UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

INGENIERÍA TÉCNICA DE TELECOMUNICACIÓN:
TELEMÁTICA

PROYECTO FIN DE CARRERA

RESOLUCIÓN DE EJERCICIOS MEDIANTE MOVIMIENTOS

Tutor: Mario Muñoz Organero
Autor: Ismael Muela Martín de Bernardo

Leganés, 15 de mayo de 2013

Agradecimientos

Agradezco a mis padres, Rafael y Petri, todo el esfuerzo que han depositado en mi educación tanto académica como en la propia vida, dandome lecciones muy importantes para mi desarrollo como persona.

A mi hermana, Noelia, por ser una gran amiga y por todo el apoyo que me ha dado a lo largo de la carrera y sobre todo en el desarrollo del proyecto final.

A mi novia, Lorena, por estar siempre ahí sin ninguna condición, ayudandome y dandome consejos en todo lo que le ha sido posible y he necesitado.

A mis abuelos por las ganas que tenian que acabara la carrera y por el animo y presión que me han dado durante ella. Acordarme especialmente de mi abuelo Alejandro al que me hubiera gustado decirle como a los demás que ya por fin terminé.

A mi tutor, Mario Muñoz, por ayudarme cada vez que me surgía una duda o problema y por haberme dado la opción de desarrollar este proyecto.

Resumen

El presente proyecto trata de explicar el diseño e implementación de una aplicación para la resolución de ejercicios, utilizando un dispositivo con sistema operativo android y el movimiento de la mano por parte de los alumnos. El dispositivo, a través de la aplicación desarrollada, será capaz de detectar el movimiento (izquierda, derecha, arriba y abajo) de la mano del alumno mediante los sensores incorporados en él, y enviar a un servidor el primer movimiento capturado.

Los ejercicios se notificarán al alumno, cuando el profesor los cree mediante una aplicación web. Trás recibir la notificación el alumno accederá al ejercicio al cual responderá realizando un gesto con la mano. Este gesto será detectado y enviado al servidor, que almacenará el movimiento en una base de datos para que el profesor, desde la aplicación web, pueda visualizar las estadísticas de cada ejercicio (número de respuestas correctas, erróneas, etc...).

Abstract

This document intends to explain the design and development of an application for resolve exercises using a mobile phone with operating system Android and hand move. Mobile will detect the movement (left, right, up, down) of the student's hand through its sensors. It will send the first move to the server.

When the teacher generated the exercise through web application, students are notified. After receiving notification, student will access to the exercise and it will respond with a gesture. This gesture will send and store to the server in a database. Teacher will view the exercise in the web application.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Motivación del proyecto

1.1. Introducción

El proyecto desarrollado, que vamos a tratar en el siguiente documento, surge de la motivación principal de realizar encuestas en la que participen los alumnos para intentar dar, más aun si cabe, dinamismo a las clases. Asimismo, ofrecer al profesor una herramienta capaz de proporcionarle información de la clase.

Para ello se ha desarrollado una aplicación para la plataforma android que permitirá al alumno participar en los ejercicios que el profesor plantea. Esta aplicación permitirá al alumno recibir los ejercicios propuestos por el profesor y responder a los ejercicios mediante un gesto con la mano. Esta respuesta será enviada al servidor a través de una conexión establecida entre el dispositivo android y dicho servidor.

Como se puede comprobar, la aplicación, tiene claramente un objetivo docente ya que pretende automatizar algunos mecanismos utilizados hoy en día en las aulas como el de "mano alzada" o el de "hoja para que firmen los asistentes a la clase". Además de proporcionar un mecanismo con el que se pueda comprobar que los alumnos se están enterando de las explicaciones pertinentes y un posible mecanismo para la evaluación continua del actual sistema universitario.

1.2. Objetivos

Los objetivos principales de este proyecto son dos:

- Proporcionar al profesor una manera automatica de realizar un seguimiento al alumno y ver su evolución y progresos en la asignatura.
- Proporcionar al alumno una manera cómoda de dar respuesta a los ejercicios propuestos por el profesor.

Debido al desarrollo de esta aplicación, el proyecto, busca conseguir los siguientes propósitos:

- Ofrecer una panorámica del mundo de los dispositivos móviles, comparando entre sí, las diferentes plataformas que existen en el mercado y analizar las posibilidades que ofrece cada plataforma al mundo del desarrollo de aplicaciones para móviles.
- Dar un detalle de la arquitectura en la que se ha desarrollado la aplicación (Android), así como, dar a conocer los detalles y las necesidades para desarrollar aplicaciones para esta plataforma.
- Explicar las distintas fases por las que ha pasado el desarrollo de la aplicación, y mostrar el nivel de investigación y desarrollo que se ha utilizado, así como el tiempo invertido y el coste del desarrollo del proyecto.
- Dar a conocer la aplicación desarrollada, exponiendo cada una de las tecnologías y procesos utilizados para tal fin.
- Exponer el modo de utilización de la aplicación, utilizando para ello un caso real, con el que se pueda explicar el modo de empleo de dicha aplicación, partiendo desde la propuesta del ejercicio del profesor hasta el envío de la respuesta por parte del alumno.

1.3. Fases del desarrollo

En esta sección se detallarán cada una de las fases por las que ha pasado la implementación de la aplicación [1]:

1. Plan operativo: En esta fase se dio forma a la idea de realizar una aplicación que sirviera para la realización de ejercicios mediante el movimiento de la mano con el objetivo de permitir al profesor realizar un seguimiento/evaluación del alumno.

2. Especificación funcional: En esta fase se propuso desarrollar una aplicación para Android, con la que los alumnos resuelvan ejercicios planteados en clase por el profesor. Estos ejercicios estarán en una diapositiva en la presentación del profesor, que abrirá el proceso de voto introduciendo una serie de parámetros (código identificativo del proceso, asignatura con la que se relacionará el proceso y la respuesta correcta del proceso) en una aplicación web. Tras esto, el proceso quedará almacenado en el sistema permitiendo al profesor lanzarlo en el momento que considere oportuno. Cuando lo lance, el sistema notificará a los alumnos en cada uno de sus terminales Android, y estos, podrán acceder al proceso de voto. La aplicación, permitirá al alumno acceder a la pantalla de voto, donde realizará un movimiento con su mano (arriba, abajo, izquierda o derecha) que, será capturado por el dispositivo y lo enviará al servidor.

3. Diseño: En esta fase veremos como satisfacer los requisitos funcionales que se proponen para la aplicación.

    - En primer lugar, se necesitará una base de datos relacional para almacenar tanto los procesos de voto creados por el profesor, como las respuestas del alumno.
    - En segundo lugar, se necesitará una aplicación web que se conecte con la base de datos y permita al profesor crear procesos, además, lanzarlos a los alumnos y realizar un seguimiento de los procesos lanzados hasta el momento podiendo visualizar las respuestas proporcionadas por los alumnos y la cantidad de respuestas de cada tipo mediante un gráfico.
    - En tercer lugar, se precisará de una aplicación Android que permita a cada alumno registrarse en el sistema, recibir notificaciones de los procesos de voto lanzados por el profesor y que detecte el movimiento gracias a los sensores existentes en el dispositivo.

4. Implementación: Esta es la fase en la que se desarrolla técnicamente cada una de las tres partes descritas en la parte de diseño.

    - Para la implementación de la base de datos se ha utilizado MySQL.
    - Para la implementación de la aplicación web se ha utilizado el lenguaje de programación Java y los componentes que ofrece este lenguaje para el desarrollo de estas aplicaciones, como son los Servlets y los JSP.
    - Para la aplicación Android se ha utilizado el lenguaje de programación Java.
    - Para la persistencia de la aplicación web se ha utilizado el API que propociona Java para tal fin, JPA. Para la persistencia de la aplicación Android, se han creado webservices en el lenguaje Java que almacenan la información en la base de datos mediante el API que proporciona Java para tal fin, JPA.

5. Integración: En esta fase se han encajado todos los componentes implementados en la fase anterior para el correcto funcionamiento de la aplicación.

6. Validación y verificación: En esta ultima fase, se han realizado pruebas de la aplicación y se ha verificado que funciona correctamente.

1.4. Medios empleados

Para la elaboración de este proyecto se ha utilizado un equipo de desarrollo que reúne las siguientes características:

- Sistema operativo: Windows 7 Profesional 64 bit
- Librerías de desarrollo: Java SDK6 y Android SDK
- Herramientas de desarrollo: IDE Eclipse JEE Juno R01 con el complemento ADT instalado

Tanto la aplicación web, como la parte servidora del proyecto, como la aplicación Android de han desarrollado en el sistema anteriormente descrito. Asimismo, se han realizado pruebas de la aplicación Android con un terminal Samsung Galaxy S3 que dispone del sistema operativo Android 4.1 (Jelly Bean).

Por último, para el desarrollo de la memoria se ha utilizado el sistema de composición de textos LaTeX.

1.5. Contenido de la memoria

En este apartado se comentará de forma breve el contenido de los capítulos del presente documento para facilitar la lectura del mismo:

- Capítulo 1: breve introducción en la que se exponen los objetivos y motivaciones del proyecto, las fases de desarrollo, o los medios empleados para llevarlo a cabo.
- Capítulo 2: comprende un análisis del estado actual de mercado de la telefonía móvil desde el punto de vista del desarrollo de software, haciendo hincapié en las posibilidades que ofrecen las distintas plataformas actuales. En este capítulo además se expone como se compone Android, su arquitectura y se explica cómo funcionan los sensores de estos dispositivos.
- Capítulo 3: Explica cómo funcionan las aplicaciones en Android, su ciclo de vida, y detalla cada uno de sus componentes.
- Capítulo 4: este capítulo comprende la fase de análisis del ciclo de desarrollo de software y se explicarán aspectos del diseño y la implementación.
- Capítulo 5: en este se explicarán las pruebas llevadas a cabo para verificar el correcto funcionamiento del sistema implementado.
- Capítulo 6: en este se explicará la historia del proyecto, las conclusiones obtenidas en su implementación y las líneas de trabajo futuras que pueden ser investigadas.
- Capítulo 7: el último capítulo de este documento contendrá la planificación seguida para el desarrollo del proyecto y un presupuesto del coste real de la aplicación.
- Apéndices: se incluyen dos apéndices donde se detallan el manual de usuario y el manual de instalación de la aplicación.
- Referencias y Bibliografía: por último se incluyen las referencias y la bibliografía consultada para la elaboración del proyecto.

Capítulo 2

Estado del arte

2.1. Introducción

Hoy en día hablar de un dispositivo movil, no es referirse a un terminal con el que se puede hablar o escribir mensajes de texto. Referirse a estos es hablar de ordenadores de bolsillo con los que se pueden realizar gran cantidad de tareas de la vida cotidiana, como por ejemplo, leer y escribir correos electrónicos o estar conectado a internet continuamente gracias a la red 3G, reproducir música, ver películas, además de permitir instalar gran cantidad de aplicaciones que nos pueden hacer más fácil el día a día.

Además, estos dispositivos, han evolucionado de una manera inconmensurable en tan solo una decada. Esta evolución es muy notoria en las caracteristicas físicas de los dispositivos moviles, como podemos comprobar en la siguiente imagen.

Figura 2.1: A la derecha Nokia 3410, comercializado en 2002. A la izquierda Samsung Galaxy S3, comercializado en 2012

Como podemos comprobar, la evolución mas notoria es la desaparición del teclado, incluido en la gran pantalla tactil (no en todos los dispositivos es de esta manera, por ejemplo, BlackBerry, cuenta con un teclado QWERTY físico). Dicha evolución no ha sido tan solo física, sino, que se les han incorporado gran cantidad de elementos que permiten prescindir del PC en algunos casos.

De hecho, los terminales móviles de hoy en día, se conocen con el nombre de "SmartPhone" o "teléfono inteligente" debido a la gran evolución que han desarrollado. A estos teléfonos, se les ha incorporado, por ejemplo, una gran variedad de sensores de posicionamiento y orientación como pueden ser el acelerómetro (permite detectar los movimientos y giros del teléfono), el GPS (permite obtener las coordenadas donde se encuentra el dispositivo) o la brújula (permite conocer hacia que punto cardinal se encuentra orientado el dispositivo).

En cuanto al software, los terminales antiguos contaban con un sistema basado en Java, mientras que los "SmartPhone" cuentan con sistemas operativos mas desarrollados (plataformas moviles), como son Android, IOS, Windows Phone o RIM. Estos sistemas operativos, permiten en la mayoría de los casos, la multitarea (permiten ejecutar varios procesos al mismo tiempo).

Figura 2.2: Distintos "SmartPhones" con sus distintos sistemas operativos

Como hemos comentado anteriormente, estos pequeños ordenadores de bolsillo, permiten la instalación de gran cantidad de aplicaciones que exprimen el rendimiento de todos los nuevos componentes con los que cuentan estos dispositivos. Para ellos, cada plataforma, cuenta con una tienda de aplicaciones especifica como son el "AppStore" de Apple, el "Play Store" de Android o el "App World" de RIM. En estas tiendas, podemos encontrar tanto aplicaciones gratuitas como de pago.

Figura 2.3: Tiendas de aplicaciones de diferentes plataformas

El "SmartPhone" es un dispositivo que también ha tenido gran penetración en la sociedad. En concreto, en España el 63 % de la población posee uno (según la Fundación Telefónica [2]). Por ello se puede considerar que el desarrollo de aplicaciones para este tipo de dispositivos puede ser una gran oportunidad de negocio.

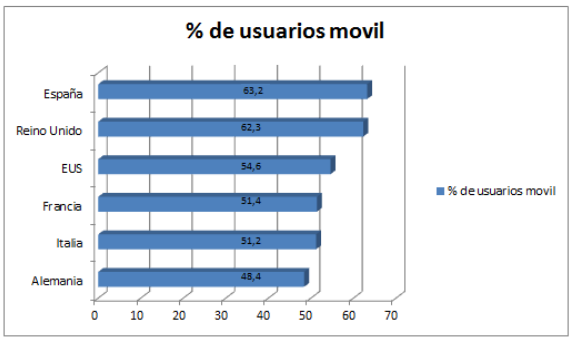

Figura 2.4: Penetración del SmartPhone en Europa. Datos de la Fundación Telefónica [2]

2.2. ¿Que es Android?

Android es un sistema operativo basado en el kernel de Linux, diseñado, inicialmente, para dispositivos móviles. Sin embargo, hoy en día, numerosos dispositivos incorporan este sistema operativo (Tablets, PCs, Reproductores Multimedia...).

Gracias a que está basado en Linux, se trata de un sistema operativo, libre, gratis y multiplataforma, por lo que, permite el desarrollo y distribución de aplicaciones de manera gratuita. [3] [4]

Para el desarrollo de aplicaciones para la plataforma Android, se utiliza el lenguaje de programación Java, aunque no utiliza la maquina virtual de Java, sino, una máquina virtual llamada Dalvik, que se encarga, de tomar los archivos generados por las clases de Java, combinarlos en uno o mas archivos ejecutables (.dex) y finalmente comprimir estos, en un paquete con extensión apk. Este paquete, será el que se instalará en el dispositivo móvil.

El propio sistema provee al desarrollador, de todas las interfaces y herramientas necesarias para el acceso a todas las funciones del dispositivo, como pueden ser, los sensores, el GPS, la agenda, etc.

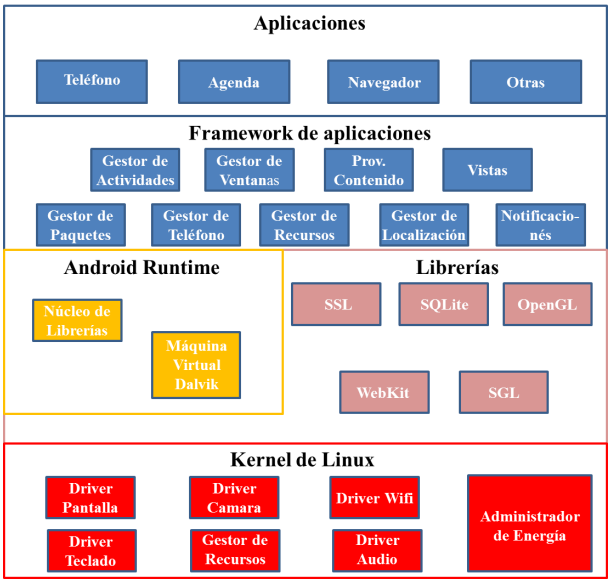

Figura 2.5: Arquitectura Android

El sistema, fue desarrollado en un principio por Android Inc, una compañía a la que absorbió Google en 2005. Hoy en día, el desarrollador del sistema en cuestión, es la Open Handset Alliance. Es una compañía liderada por Google y formada por un conglomerado de empresas con distintas funciones, por ejemplo, operadores móviles como Telefónica o Vodafone, fabricantes de terminales como Sony Ericsson o Samsung, fabricantes de semiconductores como Arm o Intel, empresas de software como Google Inc o NXP Software, empresas de comercialización como Accenture o Aplix Corporation. [5]

Figura 2.6: Firmas de la Open Handset Alliance

2.3. Evolución de Android

Android, ha evolucionado muy rápido desde su primera versión en 2008. En este punto se tratara de explicar dicha evolución y las características que se le han ido añadiendo en cada versión, asi como, las mejoras desarrolladas. Para ello, iremos viendo cada versión en detalle. [6]

⇒ Android 1.0:

Es la primera versión de Android. Sale al mercado el 23 de Septiembre de 2008, sin embargo, hasta un mes después de su salida al mercado no aparece en ningún dispositivo móvil. Fue el HTC Dream, de la empresa T-Mobile, el primer dispositivo con el sistema operativo de Google. Sin embargo, esta fue una versión muy primitiva de Android, por lo que se intento mejorar con la versión 1.1. Sin embargo, esta versión paso practicamente desapercibida. No fue así, con las posteriores versiones que detallaremos a continuación.

⇒ Android 1.5 Cupcake:

Esta versión supuso un antes y un después en el mundo de Android. Mejoro notablemente la interfaz y la compatibilidad con diferentes estándares de comunicaciones. Entre los cambios se encuentran los siguientes:

Versión	Características
1.5	• Transiciones animadas entre ventanas • Posibilidad de utilizar A-GPS • Mejoras en la velocidad del navegador Web, gracias a la inclusión de la libreria WebKit. • Personalización de los Widget de la pantalla de inicio • Inclusión del teclado en la pantalla • Posibilidad de grabar y reproducir videos • Posibilidad de copiar, pegar y buscar texto dentro una web • Mejora en la velocidad de la cámara

Cuadro 2.1: Características Android 1.5 Cupcake

⇒ Android 1.6 Donut:

En mayo de 2009 se lanzó esta versión de la plataforma. Fue la primera que se comercializo en España de la mano del terminal HTC Magic. Entre las novedades de esta versión se encuentran las siguientes:

Versión	Características
1.6	• Mejora en la velocidad de la cámara • Posibilidad de conectarse a redes VPN, 802.1x • Mejoras para controlar la batería, permitiendo ver que aplicaciones y servicios son los que mas consumen. • Nuevo motor de texto a voz.

Cuadro 2.2: Características Android 1.6 Donut

⇒ Android 2.0/2.1 Eclair:

Esta versión llegó al mercado a finales de 2009 y supuso un gran cambio en la interfaz, que pasó a ser más atractiva e introdujo los fondos animados. El dispositivo con el que llego al mercado fue el Google Nexus One, y fue la primera vez que se comparó un teléfono Android con un iPhone (el mejor del mercado hasta el momento).

Versión	Características
2.0	• Navegador con soporte de caracteristicas de HTML5. • Mejoras en la cámara (flash, zoom digital, efectos...) • Soporte para nuevos tamaños y resoluciones de pantalla. • Bluetooth 2.1. • Mejoras en el teclado virtual
2.1	• Reconocimiento de voz. • Galeria 3D. • Zoom al “pellizcar” en imagenes y navegador. • Nuevas aplicaciones. • Mejora en la duración de la batería.

Cuadro 2.3: Características Android 2.X Eclair

⇒ Android 2.2.X Froyo:

Esta versión supuso junto con Eclair un punto de inflexión en la historia de Android. Aparte de mejorar notablemente la velocidad del sistema, introdujo gran cantidad de nuevas caracteristicas importantes:

Versión	Características
2.2	• Actualizaciones automaticas de aplicaciones. • Mejora notable en el rendimiento. • Soporte para radio FM. • Soporte para Flash Player. • Nueva tienda de aplicaciones. • Soporte Wifi IEEE 802.11n.

Cuadro 2.4: Características Android 2.2 Froyo

⇒ Android 2.3 Gingerbread:

Esta versión, llego al mercado a finales del 2010 de la mano del dispositivo Samsumg Nexus S. Entre las nuevas caracteristicas se encuentran:

Versión	Características
2.3	• Soporte NFC. • Administrador de tareas para la gestión de aplicaciones. • Mejora en la gestión de energia. • Soporte nativo para telefonía VoIP. • Administrador de descargas para archivos grandes. • Soporte nativo para múltiples cámaras.

Cuadro 2.5: Características Android 2.3 Gingerbread

⇒ Android 3.X Honeycomb:

Esta versión llegó a principio de 2011 y fue diseñada totalmente orientada a tablets. Fue aquí cuando comenzó la expansión de Android más allá de los SmartPhones:

Versión	Características
3.0	• Escritorio 3D con widget rediseñados. • Sistema multitarea mejorado. • Mejora en la gestión de energia. • Mejoras en la navegación web. • Soporte para una gran cantidad de periféricos conectados vía USB. • Los widget pueden redimensionarse manualmente. • Mejora en el soporte para redes Wifi.

Cuadro 2.6: Características Android 3.0 Honeycomb

⇒ Android 4.0 Ice Cream Sandwich:

El mayor paso de Android llegó en Octubre de 2011 con la presentación de esta nueva versión junto con el dispositivo Samsung Galaxy Nexus. Supone el cambio mas radical que ha sufrido Android en su interfaz. Esta versión, unifica el uso de Android en cualquier dispositivo, ya sean teléfonos o televisores. Además aporto las siguientes características:

Versión	Características
4.0	• Nueva interfaz, mas limpia y moderna. • Multitarea mejorada. • Gestor del tráfico de datos de Internet. • Corrector de texto rediseñado y mejorado. • Desplegue de la barra de notificaciones con el dispositivo bloqueado. • Captura de pantalla pulsando encendido y volumen. • Compartir contenido entre teléfonos NFC. • Reconocimiento de voz del usuario. • Reconocimiento facial. • Reconocimiento de movimientos.

Cuadro 2.7: Características Android 4.0 Ice Cream Sandwich

⇒ Android 4.1.0 Jelly Bean:

Es la última evolución de Android hasta el momento. Posee una gran fluidez y una interfaz exquisita, además de otra serie de mejoras:

Versión	Características
4.1.0	• Mejora en la fluidez y velocidad del sistema. • Elementos ajustables automaticamente al espacio que tiene el escritorio. • Dictado por voz mejorado. • Potenciación de Android Beam (transferencia de vídeos vía NFC). • Mejora en las notificaciones para acceso a aplicaciones y galería. • Cifrado de aplicaciones para una mayor seguridad en Android. • Nuevas y mejores aplicaciones.

Cuadro 2.8: Características Android 4.1.0 Jelly Bean

2.4. Los Sensores en Android

Los sensores [7] no son más que dispositivos, que detectan una acción externa (temperatura, proximidad, presión, etc.). Estos sensores se han incorporado en los teléfonos móviles de hoy en día dando forma a lo que conocemos como Smartphone (teléfono inteligente).

Gracias a los sensores que vienen incorporados en los teléfonos móviles, nos permiten usar estos para distintas funcionalidades para las que antes necesitábamos otros dispositivos. Por ejemplo, podemos utilizar el teléfono como un navegador GPS, gracias al sensor incorporado en él, o lo podemos usar como termómetro gracias al sensor de temperatura, etc.

En concreto, Android, soporta tres categorías que son:

⇒ Sensores de movimiento: Estos sensores miden la aceleración y la rotación a lo largo de tres ejes (X, Y, Z). Algunos de ellos son:

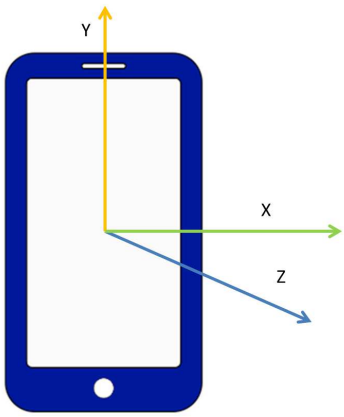

Figura 2.7: Ejes Sensores de movimiento

- Acelerómetro: Permite detectar la orientación del teléfono, para, de esta manera, adaptar el contenido de la pantalla a su orientación (vertical u horizontal). Otro uso de este sensor, puede ser en los juegos, utilizar el teléfono como un volante.

- Giroscopio: Mide la tasa de rotación del teléfono alrededor de los tres ejes. Una de las aplicaciones de este sensor es para el reconocimiento de gestos en los actuales teléfonos inteligentes.

- Vector Rotacional: Permite medir la orientación del dispositivo, proporcionando los tres elementos del vector de rotación del dispositivo.

⇒ Sensores ambientales: Como su propio nombre indica, miden parámetros ambientales, como pueden ser, la temperatura, presión, iluminación y humedad. Entre este tipo de sensores se encuentran:

- Temperatura: Mide la temperatura ambiente en grados centígrados. Se puede usar en aplicaciones, para utilizar el teléfono como un termómetro.

- Luminosidad: Mide el nivel de la luz ambiental. Este sensor consigue prolongar la batería y la vida útil del teléfono, ya que, adapta el nivel de luminosidad de la pantalla al nivel de luz ambiental.

- Presión: Permite medir la presión ambiental del aire en hPA o mbar.

⇒ Sensores de posicionamiento: Estos sensores miden la posición física del teléfono. Los sensores incluidos en esta categoría son:

- Proximidad: Este sensor permite medir la distancia a la que se encuentra el teléfono de un objeto en cm. Es, normalmente, utilizado para medir la distancia del teléfono a la oreja, para que, en el momento que este cerca de esta, bloquee la pantalla para ahorrar energía y evitar pulsaciones innecesarias.

- Brújula o magnetómetro: Mide el campo geomagnético en los tres ejes. Con él se puede conocer hacia qué punto cardinal está orientado el teléfono móvil.

- Sistema de posicionamiento global (GPS): Permite medir las coordenadas en las que se encuentra el dispositivo vía satélite. Los dispositivos más modernos incorporan A-GPS (GPS asistido) que se utiliza cuando no hay acceso inmediato a los satélites. Con este sistema, el teléfono accede a un servidor intermediario vía HTTP, en el que, se calcula el satélite más cercano al teléfono. Esto, permite una mayor velocidad de conexión con los satélites y determinar la posición del dispositivo con mayor precisión.

2.5. Comparación de Android con otras plataformas

En esta sección compararemos a Android con las demás plataformas existentes en el mercado [8], tanto a nivel de características, como a nivel de cuota de mercado, para intentar entender los motivos por los que se ha escogido el desarrollo de la aplicación a tratar en este proyecto en esta plataforma.

En la siguiente tabla vemos las características de las plataformas que actualmente están compitiendo en el mercado:

Plataforma	Android	iOS	Windows Phone	Symbian	RIM
Compañia	Open Handset Alliance	Apple	Windows	Symbian Foundation	RIM
Nucleo SO	Linux	MacOS X	Windows CE	Mobile OS	Mobile OS
Familia Procesador	ARM, MIPS, Power, x86	ARM	ARM	ARM	ARM
Licencia	Libre y abierto	Propietaria	Propietaria	Libre	Propietaria
Lanzamiento	2008	2007	2010	1997	2003
Motor Web	ARM, MIPS, Power, x86	ARM	ARM	ARM	ARM
Soporte Flash	ARM, MIPS, Power, x86	ARM	ARM	ARM	ARM
HTML5	Si	Si	Si	Si	Si
Coste Publicar Apps	25$ una vez	99$/Año	99$/Año	1$ una vez	Gratis
Plataforma de desarrollo	Windows, Linux, Mac	Mac	Windows	Windows, Linux, Mac	Windows, Mac
Soporte memoria externa	Si	No	No	Si	Si
Fabricante único	No	Si	No	No	Si
Variedad de modelos	Muy alta	Modelo único	Baja	Muy alta	Baja
Tipo de pantalla	Capacitiva, Resistiva	Capacitiva	Capacitiva	Capacitiva, Resistiva	Capacitiva, Resistiva
Aplicaciones Nativas	Si	Si	No	Si	No

Cuadro 2.9: Comparativa de plataformas

Como podemos ver, Android es el único sistema, en el que, el código es abierto y libre, por lo que, se puede obtener y modificar de manera gratuita, así como, desarrollar aplicaciones para él. Otro hándicap importante es que se puede desarrollar las aplicaciones en lenguaje Java, que es el lenguaje más practicado a lo largo de la carrera.

En el siguiente gráfico veremos la cuota de mercado a nivel mundial de las plataformas comparadas anteriormente, según un estudio de IDC[9]:

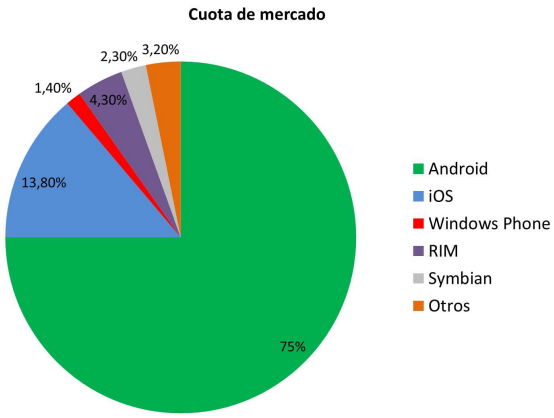

Figura 2.8: Cuota de mercado a nivel mundial de las diferentes plataformas móviles según IDC [9]

Como podemos ver, Android copa tres cuartas partes del mercado de los Smartphone. En otras palabras, tres de cada cuatro smartphones vendidos tienen como sistema operativo Android, lo que, hace que si la aplicación es desarrollada en Android se pueda utilizar en tres de cada cuatro dispositivos.

En este grafico vemos la cuota de mercado de las plataformas en comparación en España, según un estudio de Kantar World Panel[10]:

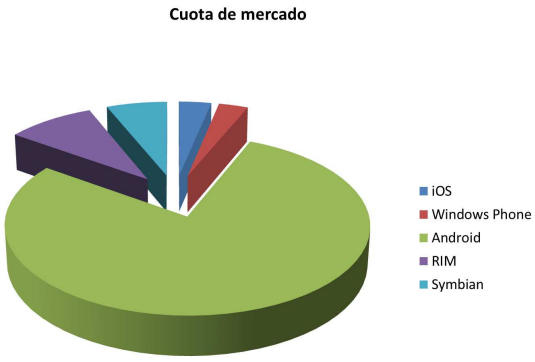

Figura 2.9: Cuota de mercado en España de las diferentes plataformas móviles según Kantar World Panel[10]

Como se puede comprobar la cuota de mercado de Android en España es muy similar a la mundial, por lo que, el motivo para elegir Android como plataforma para el desarrollo de la aplicación que trata este proyecto es el mismo.

En resumen, se puede decir, que el desarrollo en Android es gratis, y que, desarrollando para esta plataforma podemos tener la seguridad de que podrá ser instalada en una gran cantidad de dispositivos, por lo que aumentarían las posibilidades de éxito. Además, de las plataformas con más cuota de mercado (Android e iOS), los dispositivos más baratos cuentan con el sistema operativo de Google. Además existe una gran gama de diferentes modelos de Smartphone para todos los bolsillos con Android. Sin embargo, iOS, solo dispone de un único terminal en el mercado.

Capítulo 3

Aplicaciones en Android

3.1. Introducción

Las aplicaciones desarrolladas para Android [11] se dividen internamente en cuatro grandes bloques. Estos bloques, son más conocidos como componentes. Cada uno de los componentes desarrolla una función dentro de la aplicación. Los cuatro componentes de los que puede constar una aplicación Android son:

⇒ Actividades

⇒ Servicios

⇒ Receptores de difusión

⇒ Proveedores de contenido

No todas las aplicaciones, tienen porque contar con todos y cada uno de los componentes, pero, cada uno de los que posea la aplicación deben aparecer listados en un archivo denominado "AndroidManifest.xml".

En este capítulo, se verán y definirán en detalle cada uno de los componentes que forman las aplicaciones en Android, así como, el ciclo de vida de cada uno de ellos. También se tratará cómo se forma la interfaz de usuario y donde se define, en la aplicación, los permisos con los que deberá contar.

Además veremos cómo obtiene Android cada uno de los recursos (imágenes, colores, cadenas de caracteres) para mostrarlos en la interfaz de usuario de la aplicación.

Por último, se mostrará la estructura del archivo AndroidManifest.xml y se detallarán cada una de sus características.

3.2. Componentes de una aplicación

En esta sección se definirá cada uno de los componentes que puede presentar una aplicación en Android.

3.2.1. Actividades

Las actividades son los componentes más comunes de las aplicaciones en Android. Una aplicación puede constar de una o varias actividades que pueden interactuar, o no, entre sí. La actividad es mostrada al usuario como una pantalla en la que él, puede interactuar. Esta recogerá los eventos de la interacción del usuario y realizara ciertas tareas.

La mayoría de las aplicaciones se componen de varias actividades que van pasando con la interacción del usuario. En el fichero AndroidManifest.xml se define cual es la actividad principal (la que se muestra al arrancar la aplicación) y desde esta se navega hacia las demás.

Para pasar de una actividad a otra es necesario iniciar la nueva actividad. Tras esto, la actividad anterior, se pondrá en pausa y arrancara la nueva actividad. En ocasiones, es necesario, el paso de algún dato/parámetro de la actividad anterior a la nueva.

Para moverse de pantalla/actividad Android utiliza mensajes asincronos, a través, de una clase especial denominada “Intent”. Estos mensajes son mandados a los componentes mediante metodos presentes en el API de Android. Esta clase se compone de dos partes, la acción y el acto. Por ejemplo, para arrancar un navegador desde una actividad se definirá el intent del siguiente modo:

Para finalizar la navegación tendremos que ejecutar el metodos startActivity(nuevoIntent). De esta manera, la actividad será informada de que un Intent la quiere lanzar y se mostrará en la pantalla.

Para detener una actividad, es necesario, detenerla mediante metodos presentes en el API de Android, ya que, no se detienen automaticamente.

3.2.2. Servicios

Los servicios, a diferencia de las actividades, no muestran una interfaz visual al usuario y se ejecutan en segundo plano. Un buen ejemplo, es un reproductor de música que utiliza un servicio para reproducir las canciones, mientras el usuario atiende otras actividades.

Al iniciar un servicio, se presentará una interfaz, con la que, se podrá comunicar con este. En el claro ejemplo del reproductor de música, la interfaz contará con botones para pausar, parar, iniciar las canciones.

La inicialización de estos componentes, al igual que las actividades se realiza mediante Intent. Cuando se requiera detener un servicio es necesario hacerlo manualmente mediante metodos presentes en el API de Android ya que no se detienen automaticamente.

3.2.3. Proveedores de contenido

Las aplicaciones, pueden almacenar datos en ficheros, una base de datos SQLite o en preferencias (SharedPreferences). Un proveedor de contenido, permite compartir los datos almacenados de una aplicación con otras aplicaciones.

Para inicializar estos basta con hacer una consulta a los datos que contienen y se mantendrán activos mientras atiendan la consulta en cuestión. Por lo tanto este tipo de componente no ha de ser desactivado en ningún momento.

3.2.4. Receptores de difusión

Los receptores de difusión, se utilizan, cuando queremos que la aplicación reacciones a un evento externo. El ejemplo más claro, es que el teléfono suene cuando se reciba una llamada o que muestre una notificación cuando se recibe un mensaje.

Estos componentes no disponen de interfaz gráfica, pero, pueden invocar actividades en respuesta al mensaje recibido o, como hemos comentado anteriormente, utilizar el gestor de notificaciones para informar al usuario del evento ocurrido.

Para lanzar estos componentes no es necesario que la aplicación que los contenga se esté ejecutando. Cuando se reciba la llamada al receptor de difusión, el propio sistema se encargará, si es necesario, de abrir la aplicación. Este componente permanecerá activo mientras atiende la consulta.

3.3. Ciclo de vida de los componentes

En esta sección se mostrarán los principales aspectos del ciclo de vida de cada uno de los componentes que forman una aplicación.

3.3.1. Actividad

El ciclo de vida de una actividad, se refiere, a los estados por los que va pasando la actividad desde que se crea hasta que se destruye. En el siguiente diagrama podemos ver cada uno de los estados que va atravesando la actividad. A continuación definiremos cada uno de los métodos que componen el ciclo de vida.

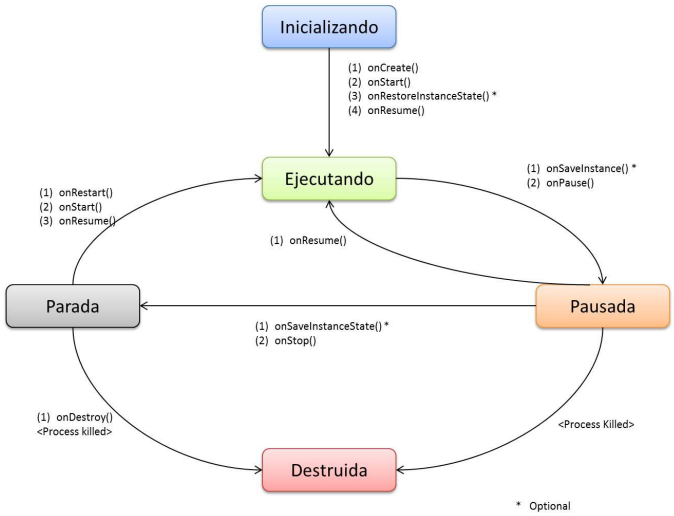

Figura 3.1: Ciclo de vida de una actividad android

Como podemos comprobar, una actividad, cuenta con cuatro estados que son:

⇒ Activa (Ejecutando): Cuando la actividad está la primera en la pila de ejecución, es decir, la actividad se está mostrando y el usuario puede interactuar con ella.

⇒ Pausada: La actividad se encuentra en segundo plano, pero aún está visible. Está por debajo de otra actividad pero no está tapada del todo. En este caso, el sistema puede cerrar la actividad en caso de necesitar liberar recursos.

⇒ Parada: Ha pasado a segundo plano y está completamente tapada por la nueva actividad. En este caso la actividad también puede ser cerrada por el sistema para liberar recursos.

⇒ Destruida: La actividad ya no está disponible. Se han liberado completamente sus recursos y si se requiere abrirla de nuevo se necesitará generar un nuevo ciclo de vida para ella.

La transición entre los distintos estados se realiza con los siguientes métodos:

⇒ onCreate(): Se llama al crear la actividad. Es el que genera la interfaz gráfica de la pantalla. La actividad no puede ser destruida tras la ejecucion de este método. Posteriormente se ejecuta el método onStart().

⇒ onRestart(): Se lanza justo antes de que una actividad que estaba parada vuelva a estar activa. Tras esto se llama al método onStart() y la actividad no puede ser destruida.

⇒ onStart(): Se ejecuta justo antes de que la aplicación sea visible para el usuario. El siguiente método que se ejecutará será onResume u onStop().

⇒ onResume(): Se ejecuta justo antes de que el usuario pueda interactuar con la actividad. El siguiente método que se ejecute será onPause().

⇒ onPause(): Este método se llamara cuando la actividad vaya a ser tapada con otra, es decir, cuando se ejecute el método onRestart() de otra. En este método se liberará todo lo que consuma recursos de la actividad y no se crearán procesos largos en el, ay que, no se podrá ejecutar el método onResume() de la otra actividad, hasta que no finalice la ejecución de este. El siguiente método que se ejecutará será onResume() u onStop().

⇒ onStop(): Se ejecuta cuando la actividad se hace invisible para el usuario, bien porque ha sido tapada por otra (a continuación se ejecutará onRestart()), o bien porque la actividad haya sido destruida invocando al método onDestroy().

⇒ onDestroy(): Se llama justo antes de destruir la actividad. Durante la destrucción de la actividad se perderán todos los datos asociados a ella, por lo que es conveniente, tener algún tipo de control en este método de los datos que no se quieran perder.

Vistos los métodos que controlan el ciclo de vida de una actividad podemos diferenciar, según la ejecución de estos, tres bucles de ejecución:

⇒ Ciclo de vida completo: Va desde la llamada al método onCreate() donde se genera la actividad, hasta la llamada al método onDestroy() donde el sistema libera todos recursos que utiliza la aplicación.

⇒ Tiempo de vida visible: ocurre entre la llamada al método onStart(), hasta la ejecución del método onStop(). En este tiempo, el usuario, puede ver la actividad y el sistema aún guarda los datos de la actividad (no libera los recursos), aunque podría liberar los recursos si la aplicación no se encontrará en primer plano.

⇒ Tiempo de vida en primer plano: Va desde la llamada al onCreate(), hasta la llamada al onPause(). Es el tiempo que la actividad se mantiene en primer plano y que, por tanto, el usuario podrá interactuar con ella.

Como hemos comentado en algunos puntos de esta sección, el sistema, puede en algunos casos, eliminar los datos referentes a la actividad para liberar recursos. Por ello, es necesario, mantener los datos que se desean guardar cuando la aplicación no está en primer plano. Para esto, el API de android, ofrece ciertos métodos y objetos para almacenar estos datos.

3.3.2. Servicio

Los servicios pueden ser ejecutados de dos maneras diferentes, por lo que, su ciclo de vida será dependiente de la forma en que se realice su ejecución.

⇒ Ejecución automática: El servicio es iniciado con el método startService(), y, se mantiene vivo en segundo plano, hasta que sea

detenido desde algún punto de la aplicación, o, sea detenido por el sistema mediante el método stopService(). El diagrama de este flujo de ejecución del servicio es el siguiente:

Figura 3.2: Ciclo de vida de un servicio 1

⇒ Ejecución programada: Los servicios pueden ser lanzados de manera programática a través de una interfaz que el exportará el servicio, con la que se podrá interactuar con él, mediante el método bindService() y detenidos mediante el método unbindService(). El diagrama de este flujo de ejecución es el siguiente:

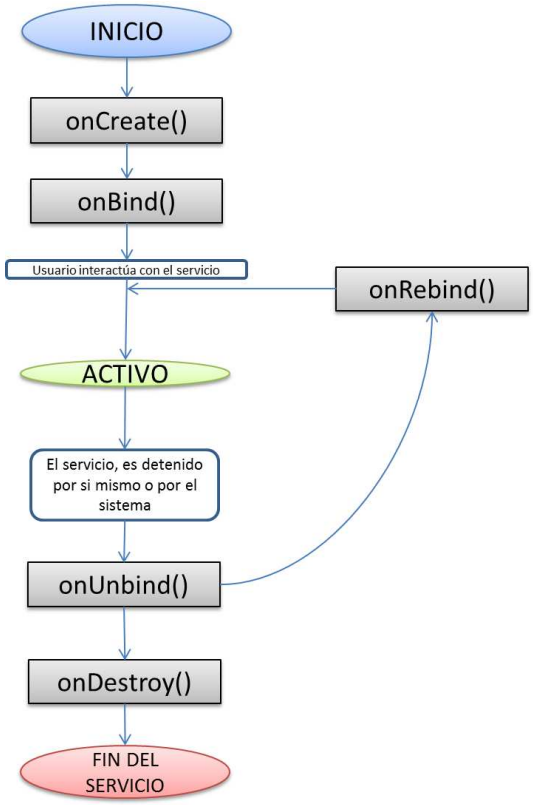

Figura 3.3: Ciclo de vida de un servicio 2

Como se puede comprobar viendo los diagramas, los servicios, al igual que las actividades, se componen de unos métodos que definen su ciclo de vida. Fijándonos en ellos se pueden definir dos flujos de trabajo bien diferenciados:

⇒ Ciclo de vida completa: Va desde la ejecución del método onCreate(), hasta la ejecución del método onDestroy().

⇒ Ciclo de vida activo: Comienza con la llamada al método onStart().

También, se puede visualizar, que el método onStart(), solamente es ejecutado cuando el servicio se lanza de la primera forma explicada. Sin embargo existen otros métodos para controlar al servicio cuando este, permite el acceso a el desde otro componente. Estos métodos son onBind(), onUnbind() y onRebind().

3.3.3. Proveedor de contenido

Al igual que los receptores de difusión, su ciclo de vida también es muy corto. De hecho, los proveedores de contenido, solamente permanecen

activos, cuando otro componente realiza una consulta a los datos que este contiene.

3.3.4. Receptor de difusión

Los receptores de difusión tienen un ciclo de vida muy corto. De hecho, tienen solamente un método asociado a su ciclo de vida, onReceive(Context contexto, Intent mensaje). De esta manera, cuando un mensaje llega a un receptor de difusión, ejecuta este método pasándole el mensaje recibido. Se considera que el receptor de difusión está activo solamente mientras este método se está ejecutando, y por tanto, no podrá ser eliminado de la memoria del sistema para liberar recursos.

3.4. Interfaz de usuario

La interfaz de usuario en Android, tiene dos maneras de ser desarrollada. La primera es mediante código Java y la segunda e ideal por el modelo-vista-controlador que indica la necesidad de separar la interfaz de usuario de la lógica del programa, es mediante uno o varios ficheros XML donde se definirán cada uno de los componentes que forman la interfaz.

De esta manera, la interfaz de usuario, estará expresada en los ficheros XML en forma de árbol como muestra el siguiente ejemplo:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World"/>
</LinearLayout>
```

Figura 3.4: Ejemplo XML Interfaz de usuario

3.4.1. Jerarquía de los componentes de la interfaz

La interfaz de usuario de android se construye con dos componentes que son los View y los ViewGroup.

⇒ View: Estos componentes, gestionan y almacenan información, acerca del contenido y la disposición de una porción concreta de la pantalla. Gestionan aspectos como el tamaño, el foco de atención, o la interacción del usuario con esa porción. Sirven de base para otros objetos llamados widgets que ofrecen funcionalidades ya implementadas y que sirven para la interacción del usuario con la pantalla. Alguno de estos widgets son los botones, campos de texto, scroll...

⇒ ViewGroup: Sirven para definir la disposición o layout de los componentes que forman la interfaz.

Para definir la interfaz de usuario, es necesario establecer una relación jerárquica entre estos dos componentes como la que se muestra a continuación:

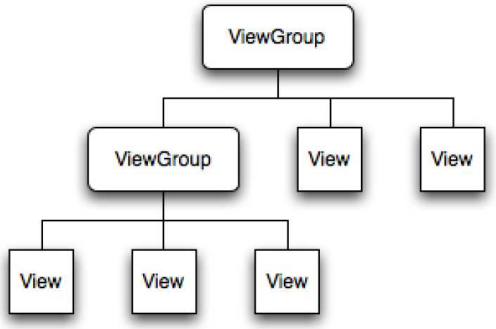

Figura 3.5: Estructura interfaz de usuario

Los elementos View pueden realizar una serie de acciones al interactuar con el usuario. Para controlar estas acciones disponemos de dos formas:

1. Estableciendo métodos dentro de cada objeto View que se ejecuten después de la interacción del usuario. Por ejemplo, al pulsar un botón se lanzara el método onClickListener() que realizará la acción que se le indique.

2. Redefiniendo los eventos desencadenados por la acción del usuario al implementar un nuevo objeto View.

Otro elemento importante de la interfaz de usuario es el menú contextual, que se abre al pulsar el botón que disponen todos los dispositivos android para tal fin. El menú se define de manera diferente a la interfaz de usuario, ya que, se hace dentro de la actividad en los métodos onCreateContextMenu() y onCreateOptionsMenu(). En estos métodos, se especifican los elementos que forman parte de cada menú, siendo el sistema, el encargado de formar la estructura jerarquica necesaria para mostrar el menú. Además, estos componentes manejan sus propios eventos, por lo que, no es necesario establecer métodos para controlarlos.

3.5. Obtención de recursos

Los recursos son archivos externos que son utilizados por los componentes de las aplicaciones (imágenes, textos). Android soporta un gran número de formatos de ficheros que pueden ser utilizados como recursos (png, jpg, xml, etc).

Para que los componentes puedan referenciar correctamente estos recursos, es necesario, organizarlos en una estructura de directorios dentro de la aplicación, debido a que, por ejemplo, las imágenes deben estar en distinto directorio que los strings. La organización de los directorios es la siguiente:

⇒ /res/layout/ – Archivos xml de las interfaces de usuario.

⇒ /res/drawable/ – Imágenes para iconos, fondos, etc.

⇒ /res/animations/ – Animaciones

⇒ /res/values/ – Ficheros xml con estilos, strings y arrays.

⇒ /res/raw/ – Archivos multimedia, videos, mp3, etc.

Los recursos se pueden también organizar de tal manera, para que, la aplicación los utilice en función de la configuración de idioma o hardware del dispositivo. Para crear recursos dependientes de configuraciones, se crean las carpetas con el mismo nombre separado por un guion del parámetro de configuración que utilizaremos. Por ejemplo, para configurar los strings en distinto idioma crearemos los siguientes directorios:

⇒ /res/values-es/strings.xml

⇒ /res/values-en/strings.xml

3.6. Archivo Manifest.xml

Es un archivo XML con el que debe contar toda aplicación Android. Es muy importante puesto que en él se describen los siguientes puntos:

⇒ Definir el nombre del paquete APK que sirve como identificador único de la aplicación.

⇒ Describe cada uno de los componentes de la aplicación (actividades, proveedores de contenido, receptores de difusión, servicios). Se indican en el las clases que definen cada uno de los componentes y si son el componente principal o no.

⇒ Se definen en el los permisos con los que será necesario dotar a la aplicación a la hora de instalarla en el dispositivo móvil.

⇒ Se definen en el los permisos que debe poseer otra aplicación para usar componentes de la suya si se diera el caso.

⇒ Define la versión minima del API de Android que necesita la aplicación para funcionar.

Capítulo 4

Analisis, diseño e implementación de la aplicación

4.1. Introducción

Una vez comprendidos y asimilados los conceptos basicos de android y de sus perifericos (los sensores), se procederá a detallar el analisis, diseño e implementación de la aplicación a la que hace referencia esta memoria.

En primer lugar, se detallarán las necesidades de usuario que tendrán que resolver la aplicación. Posteriormente, se planteará el diseño de la aplicación para los requisitos de usuario recogidos. Finalmente, se expondrán los aspectos mas relevantes del desarrollo de la aplicación.

4.2. Analisis

En este apartado se detallarán las necesidades de usuario con las que tendrá que contar la aplicación. Debido a que la aplicación tendrá dos tipos diferenciados de usuarios, dividiremos las necesidades en dos grupos que serán profesores y alumnos. Se detallarán también las restricciones con los que contará dicha aplicación.

4.2.1. Necesidades previas

Se mostrarán, a continuación, los requisitos previos necesarios antes de empezar a trabajar con la aplicación, en distintas tablas con el siguiente formato:

- PREV_X: Identificador univoco del requisito previo, siendo X un valor numérico que comenzará en uno.
- Descripción: Detalle del requisito previo.
- Criticidad: Nivel de necesidad del requisito dentro del proyecto. Toma los valores del 1 al 3, siendo el 1 criticidad baja, 2 criticidad intermedia y 3 criticidad alta.

Identificador	PREV_1: Base de datos
Descripción	Existirá una base de datos desarrollada en MySQL donde se almacenarán todos los datos necesarios para el correcto funcionamiento de la aplicación.
Criticidad	ALTA

Cuadro 4.1: Necesidad PREV_1

Identificador	PREV_2: Profesores dados de alta
Descripción	Los profesores que puedan utilizar la aplicación, estarán dados de alta en una tabla de la base de datos.
Criticidad	ALTA

Cuadro 4.2: Necesidad PREV_2

4.2.2. Necesidades del profesor

Se mostrarán, a continuación, las necesidades del profesor en varias tablas que contendrá el siguiente formato:

- PROF_X: Identificador univoco de la necesidad, siendo X un valor numérico que comenzará en uno.
- Descripción: Detalle de la necesidad del profesor.
- Criticidad: Nivel de necesidad del requisito dentro del proyecto. Toma los valores del 1 al 3, siendo el 1 criticidad baja, 2 criticidad intermedia y 3 criticidad alta.

Identificador	PROF_1: Crear Procesos de Voto
Descripción	El profesor podrá crear procesos de voto
Criticidad	ALTA

Cuadro 4.3: Necesidad PROF_1

Identificador	PROF_2: Asignar asignatura a proceso de voto
Descripción	El profesor podrá asignar una asignatura a cada procesos de voto
Criticidad	ALTA

Cuadro 4.4: Necesidad PROF_2

Identificador	PROF_3: Almacenar Procesos de Voto
Descripción	El profesor podrá almacenar procesos de voto
Criticidad	ALTA

Cuadro 4.5: Necesidad PROF_3

Identificador	PROF_4: Lanzar Procesos de Voto
Descripción	El profesor podrá lanzar a los alumnos de la asignatura cada uno de los procesos de voto almacenados
Criticidad	ALTA

Cuadro 4.6: Necesidad PROF_4

Identificador	PROF_5: Visualizar las respuestas de los alumnos
Descripción	El profesor podrá ver las respuestas de cada alumno a cada proceso para realizar un seguimiento de este
Criticidad	ALTA

Cuadro 4.7: Necesidad PROF_5

Identificador	PROF_6: Ver estadística del proceso
Descripción	El profesor podrá visualizar un gráfico para ver los resultados del proceso
Criticidad	MEDIA

Cuadro 4.8: Necesidad PROF_6

Identificador	PROF_7: Desactivar procesos de voto
Descripción	Los procesos de voto que el profesor lance solo permanecerán activos durante un minuto para que los alumnos envíen el voto.
Criticidad	ALTA

Cuadro 4.9: Necesidad PROF_7

Identificador	PROF_8: Gestión de la base de datos
Descripción	El profesor podrá dar de alta asignaturas, alumnos y profesores en la base de datos desde la aplicación web. Los alumnos podrán ser relacionados con tantas asignaturas como existan en la base de datos, además, de poderse modificar dichas asignaciones
Criticidad	ALTA

Cuadro 4.10: Necesidad PROF_8

4.2.3. Necesidades del alumno

Se mostrarán, a continuación, las necesidades del alumno en varias tablas que contendrá el siguiente formato:

- ALUM_X: Identificador univoco de la necesidad, siendo X un valor numérico que comenzará en uno.
- Descripción: Detalle de la necesidad del alumno.
- Criticidad: Nivel de necesidad del requisito dentro del proyecto. Toma los valores del 1 al 3, siendo el 1 criticidad baja, 2 criticidad intermedia y 3 criticidad alta.

Identificador	ALUM_1: Registro en la aplicación
Descripción	El alumno podrá registrarse en la aplicación, para recibir los procesos en los que puede votar
Criticidad	ALTA

Cuadro 4.11: Necesidad ALUM_1

Identificador	ALUM_2: Configuración de la dirección del servidor
Descripción	El alumno podrá configurar la dirección del servidor de procesado de datos
Criticidad	ALTA

Cuadro 4.12: Necesidad ALUM_2

Identificador	ALUM_3: Recibir notificación de los procesos
Descripción	El alumno recibirá una notificación cada vez que el profesor lance un proceso de voto
Criticidad	MEDIA

Cuadro 4.13: Necesidad ALUM_3

Identificador	ALUM_4: Votar en cada proceso mediante movimiento
Descripción	El alumno podrá enviar su voto mediante un movimiento de la mano
Criticidad	ALTA

Cuadro 4.14: Necesidad ALUM_4

Identificador	ALUM_5: Almacenamiento del usuario y la IP del servidor
Descripción	El alumno sólo tendrá que registrarse cuando instale la aplicación en el dispositivo. La aplicación deberá disponer de memoria
Criticidad	ALTA

Cuadro 4.15: Necesidad ALUM_5

Identificador	ALUM_6: Busqueda de procesos de voto
Descripción	El alumno, en caso de no recibir notificación, podrá buscar procesos de votos en los que esté involucrado
Criticidad	ALTA

Cuadro 4.16: Necesidad ALUM_6

4.2.4. Restricciones de la aplicación

Se mostrarán, a continuación, las restricciones con las que contará la aplicación en varias tablas que contendrán el siguiente formato:

- RES_X: Identificador univoco de la restricción, siendo X un valor numérico que comenzará en uno.
- Descripción: Detalle del requisito de restricción.
- Criticidad: Nivel de necesidad del requisito dentro del proyecto. Toma los valores del 1 al 3, siendo el 1 criticidad baja, 2 criticidad intermedia y 3 criticidad alta.

Identificador	RES_1: Conectividad
Descripción	Deberá existir conectividad a Internet mediante wifi para poder utilizar la funcionalidad de la aplicación.
Criticidad	ALTA

Cuadro 4.17: Restricción RES_1

Identificador	RES_2: Direcciones IP
Descripción	Las direcciones IP deberán tener el formato IPv4, es decir, X.Y.W.Z (siendo cada letra un número comprendido entre 0 y 255).
Criticidad	ALTA

Cuadro 4.18: Restricción RES_2

Identificador	RES_3: Portabilidad
Descripción	Tanto la aplicación del profesor, como la parte servidora, se podrán instalar en distintos sistemas operativos (Windows, Linux o Mac) ya que, estarán desarrolladas en Java que es un lenguaje multiplataforma.
Criticidad	ALTA

Cuadro 4.19: Restricción RES_3

Identificador	RES_4: Idioma de la interfaz
Descripción	Tanto la interfaz del profesor como la del alumno se mostrarán en español.
Criticidad	MEDIA

Cuadro 4.20: Restricción RES_4

Identificador	RES_5: Plataforma Android
Descripción	La aplicación del alumno podrá instalarse en terminales que posean una versión de Android 2.2 o superior.
Criticidad	ALTA

Cuadro 4.21: Restricción RES_5

Identificador	RES_6: Permisos
Descripción	Al instalar la aplicación en el dispositivo móvil, el alumno, deberá aceptar una serie de permisos para el correcto funcionamiento de la aplicación.
Criticidad	ALTA

Cuadro 4.22: Restricción RES_6

Identificador	RES_7: Alta de profesores
Descripción	El alta de profesores se realizará directamente en la base de datos.
Criticidad	ALTA

Cuadro 4.23: Restricción RES_7

4.3. Diseño

Esta aplicación se dividirá en dos partes, la aplicación de los alumnos y la de los profesores. En este apartado se hablará del diseño software de la aplicación. Se detallarán los aspectos relacionados con el diseño de la base de datos, así como, el diseño arquitectónico de la plataforma y el diseño de cada una de las interfaces de usuario.

4.3.1. Diseño de la Base de Datos

Ambas aplicaciones compartirán una base de datos relacional que seguirá el modelo E/R o entidad relación. Dicha base de datos será modelada a partir de un diagrama en el que se detallarán cada una de las entidades, así como, las relaciones entre estas y las propiedades y atributos de cada una.

Modelo Entidad-Relación

Las entidades son los objetos que recogen información y serán representadas, en el diagrama, como rectángulos, dentro del cual, aparecerá el nombre de cada entidad.
Las relaciones son las asociaciones entre una o varias entidades. Dichas relaciones se representan mediante rombos indicando dentro de él, la función que realiza la relación. Existen tres tipos de relaciones:

- 1 a 1: Las entidades que asocie esta relación solo figurarán una vez.
- 1 a N o N a 1: Una entidad puede figurar N veces mientras que la otra solo figurará una vez.
- N a M: Las entidades que relacionen pueden figurar N veces cada una.

Las entidades y relaciones, estarán formadas por atributos. Cada uno de estos representa una característica de la entidad o de la relación de varias entidades. Se representan mediante círculos que cuelgan de cada una de las entidades o relaciones.

El diagrama de la base de datos de la aplicación quedará del siguiente modo:

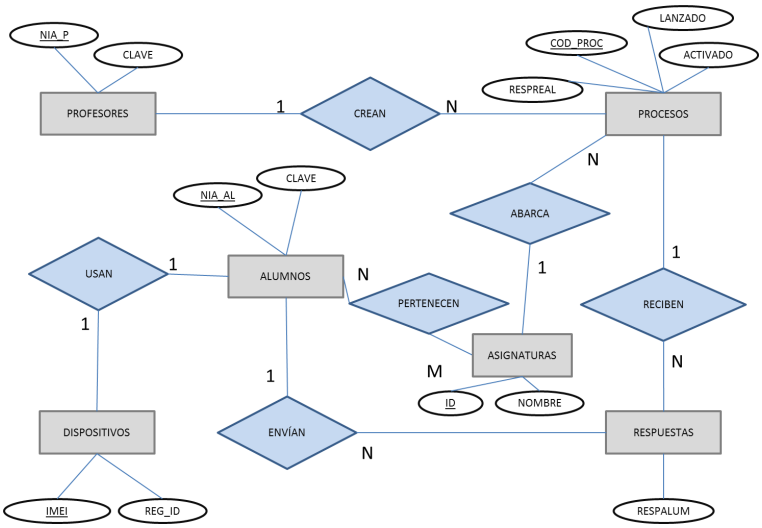

Figura 4.1: Diagrama Entidad - Relación

Como se puede comprobar en la ilustración anterior, existirán diferentes PROCESOS que serán gestionados por los PROFESORES. Estos, serán emitidos a los alumnos a los que el profesor haga participes de cada uno. Cuando los ALUMNOS reciban el proceso, emitirán una RESPUESTA. Cada proceso puede tener de 1 a N respuestas.

Cada alumno recibirá de 1 a N procesos y emitirá de 1 a N respuestas en función de la cantidad de procesos recibidos. Estas respuestas se enviarán por parte de los alumnos desde los DISPOSITIVOS. Cada alumno podrá dar de alta un dispositivo con el que emitir respuestas.

Modelo Relacional

En este modelo todos los datos son almacenados en relaciones. Cada relación será un conjunto de datos donde existirá toda la información que se podrá recuperar o administrar mediante consultas.

Cada una de las relaciones representará una tabla y cada atributo de la relación será un campo de la tabla. De esta manera, cada tabla, estará formado por un conjunto de filas o tuplas.
A partir del modelo entidad relación del apartado anterior, se puede pasar al modelo de base de datos relacional del siguiente modo:

- Por cada entidad del modelo entidad relación se creará una tabla del modelo relacional.
- Por cada relación N:M del modelo entidad relación se creará una tabla que contendrá cada una de las claves primarias de las entidades que relaciona que compondrán la clave primaria de la tabla y se comportarán también como claves ajenas, así como, los atributos de la relación.
- Por cada relación 1:N se añadirá a la tabla del lado N la clave primaria de la tabla del lado 1 y se comportara como clave ajena.
- Por cada relación 1:1 se añadirá a la tabla del lado que se considere oportuno la clave de la tabla del otro lado de la relación y se comportará como clave ajena.

De esta manera, el modelo relacional de la aplicación que se está tratando quedará del siguiente modo:

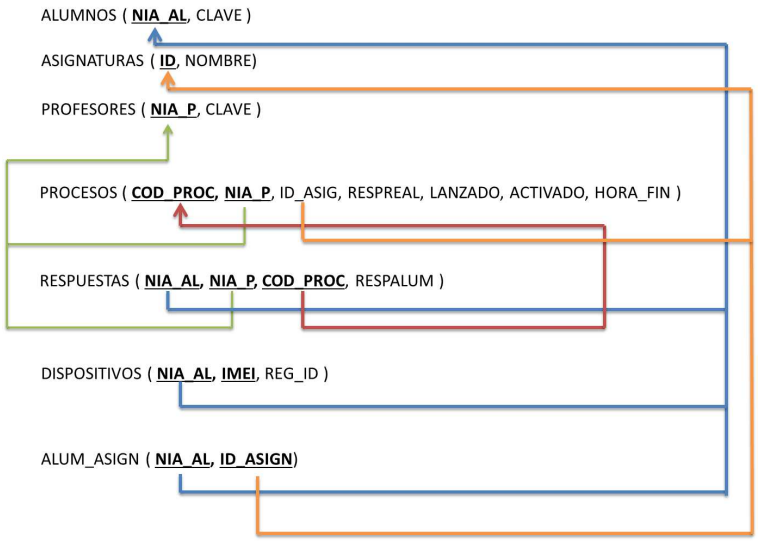

Figura 4.2: Diagrama Relacional

- ALUMNOS: Esta tabla contendrá cada uno de los alumnos que podrán darse de alta en la aplicación que se realizará para enviar el voto desde los dispositivos móviles. Los alumnos que no se encuentren dado de alta en esta tabla no podrán hacer uso de la aplicación. Esta tabla dispondrá de dos campos que son:
  - NIA_AL: Será el identificador universitario de cada alumnos. El usuario que utilizan para acceder a todas las aplicaciones de la Universidad. Este campo será la clave primaria de la tabla. Será de tipo VARCHAR con un máximo de 100 caracteres.
  - CLAVE: Será la contraseña que cada alumno utilice en la Universidad. Será de tipo VARCHAR con un máximo de 12 caracteres.
- PROFESORES: Esta tabla contendrá cada uno de los profesores que podrán utilizar la aplicación web para la creación de procesos de voto. Los profesores que no se encuentren dado de alta en esta tabla no podrán hacer uso de la aplicación. Esta tabla dispondrá de dos campos que son:
  - NIA_P: Será el identificador universitario de cada profesor. El usuario que utilizan para acceder a todas las aplicaciones de la Universidad. Este campo será la clave primaria de la tabla. Será de tipo VARCHAR con un máximo de 100 caracteres.
  - CLAVE: Será la contraseña que cada profesor utilice en la Universidad. Será de tipo VARCHAR con un máximo de 12 caracteres.
- ASIGNATURAS: Esta tabla contendrá cada una de las asignaturas que podrán ser asignadas a los alumnos y a las que se le podrán asignar procesos de voto. Esta tabla dispondrá de dos campos que son:
  - ID: Será el identificador de la asignatura. Su tipo sera INTEGER y crecerá de forma automatica según se vayan introduciendo asignaturas en la base de datos.
  - NOMBRE: Será el nombre identificativo de la asignatura. Su tipo será un VARCHAR de 50 posiciones.
- PROCESOS: Esta tabla contendrá cada uno de los procesos que los profesores crearán para que los alumnos emitan su voto. Como clave primaria tiene el COD_PROC que será un código que identificará a cada proceso, además del NIA_P que es el identificador del profesor que crea el proceso. El NIA_P además actuará también como clave ajena que es la clave primaria de la tabla de profesores. Procesos estará relacionada

con profesores puesto que estos son los que crean los procesos. Los campos de los que dispondrá la tabla serán:

- COD_PROC: Será un código que identifique al proceso. Junto con el NIA_P formarán la clave primaria de la tabla. El tipo será VARCHAR de 10 caracteres como máximo.
- NIA_P: Mismo campo que en la tabla de profesores.
- ID_ASIGN: Será la asignatura con la que este relacionado el proceso y de la que extraerá los alumnos para realizar la notificación del proceso. No podrán existir ID_ASIGN que no aparezcan en la tabla de asignaturas.
- RESPREAL: Será la respuesta correcta del proceso. El tipo será un CHAR de 1, pudiendo tomar los valores 'L'(izquierda), 'R'(derecha), 'U'(arriba), 'D'(abajo).
- LANZADO: Será un flag que se activará cunado el profesor lance el proceso a los alumnos. Los valores que puede tomar son 'X' o vacío.
- ACTIVADO: Será un flag que se marcará con 'X' cuando el proceso esté disponible para que los alumnos envíen su voto. Solamente se marcará con 'X' durante el tiempo de vida del proceso.
- HORA_FIN: Sera un parametro de tipo Time. En el, se guardará la hora en la que se ha lanzado el proceso mas un minuto.

◦ RESPUESTAS: Esta tabla contendrá la relación de respuestas de cada alumno para cada proceso. Su clave primaria la formarán los campos NIA_AL (clave ajena de la tabla de alumnos), NIA_P (clave ajena de la tabla de profesores) y COD_PROC (clave ajena de la tabla de procesos). Los campos de los que dispondrá la tabla son los siguientes:

- NIA_AL: Mismo campo que en la tabla de alumnos.
- NIA_P: Mismo campo que en la tabla de profesores.
- COD_PROC: Mismo campo que en la tabla de procesos.
- RESPALUM: Será la respuesta enviada por cada alumno. El tipo será un CHAR de 1, pudiendo tomar los valores 'L'(izquierda), 'R'(derecha), 'U'(arriba), 'D'(abajo).

◦ DISPOSITIVOS: Será la tabla que almacene la relación de alumnos con sus respectivos dispositivos, para poder identificarlos a la hora de votar. Como clave primaria tendrá el campo NIA_AL (clave ajena de la tabla de alumnos) y el campo IMEI.

- NIA_AL: Mismo campo que en la tabla de alumnos.
- IMEI: Será el identificador único de cada dispositivo para votar. Se almacenará en la base de datos en el momento en que un alumno acceda con su usuario y clave a la aplicación para enviar un voto.
- REG_ID: Será el identificador del dispositivo proporcionado por Google para el envío de notificaciones PUSH. Será un campo de tipo TEXT y será guardado en la base de datos en el momento que un alumno acceda con su usuario y clave a la aplicación para enviar un voto.

◦ ALUM_ASIGN: Esta tabla almacenará la relación de cada asignatura con cada alumno. Como clave primaria tendrá el ID_ASIGN(clave ajena de la tabla de asignaturas) y NIA_AL(clave ajena de la tabla de alumnos):

- NIA_AL: Mismo campo que en la tabla de alumnos.
- ID_ASIGN: Mismo campo que en la tabla de asignaturas.

◦ DESACTIVA: Este evento se ejecutará cada segundo y comprobará el campo HORA_FIN de la tabla PROCESOS de la base de datos. En caso de que esta hora sea menor que la actual desactivará el proceso para que no se puedan enviar mas votos del mismo.

4.3.2. Aplicación del profesor

Para la creación de procesos de voto y su correspondiente seguimiento por parte de los profesores se ha desarrollado una aplicación web cuyo diseño se va a explicar en este apartado.

Arquitectura de sistemas

La aplicación del profesor deberá almacenar los procesos creados en la base de datos, así como, leerlos para su posterior tratamiento. Para ello utilizará el conector de MySQL "mysql-connector-java-5.1.18-bin.jar" para establecer las conexiones con la base de datos. La arquitectura del sistema quedará del siguiente modo:

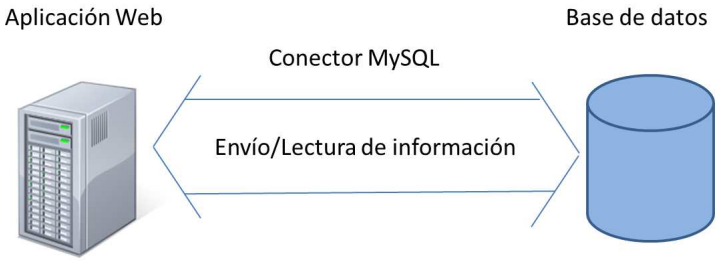

Figura 4.3: Arquitectura de sistemas

Como se puede ver en la imagen anterior la aplicación web se conectará con la base de datos para almacenar y recoger la información que necesite en cada momento. Para ello se implementarán distintas interfaces que permitan realizar dichas acciones. A continuación se detallarán cada uno de los módulos de la aplicación necesarios para conseguir el correcto funcionamiento e integración de la aplicación web con la base de datos.

Arquitectura de la aplicación

En esta sección se detallará la arquitectura de componentes y el uso de cada uno dentro de la aplicación web a implementar. Para ello, se empezará mostrando un diagramos con la arquitectura de componentes de la aplicación.

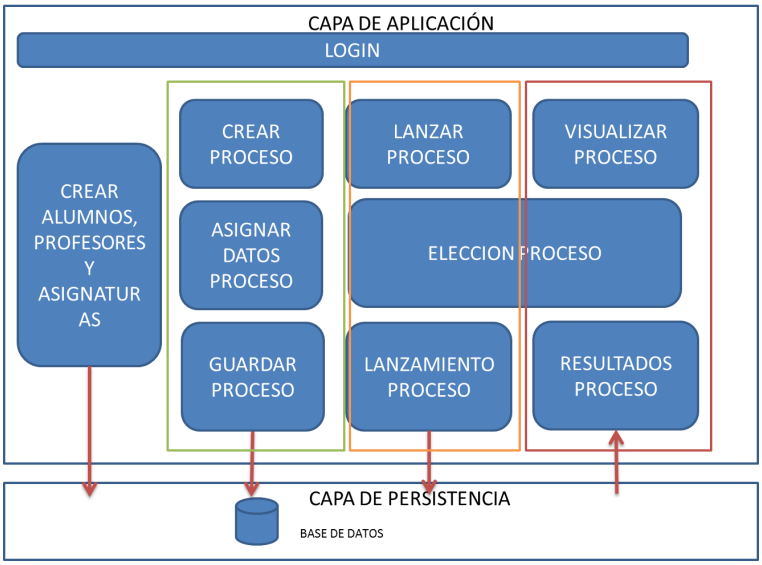

Figura 4.4: Arquitectura de componentes

Como se puede comprobar la aplicación de dividirá en cuatro grandes módulos que serán: gestión de la base de datos (creación de asignaturas, creación y modificación de alumnos y profesores), creación de proceso, lanzamiento de proceso y visualización de proceso. Todos los módulos de la aplicación accederán a la base de datos para almacenar o extraer información. Además de estos módulos se dispondrá de la interfaz de usuario que será la parte que interactúe con el usuario para navegar por la aplicación y crear/lanzar procesos, así como, gestionar y visualizar los procesos antiguos.

Definición de los componentes de la aplicación

- Módulo de login: Este módulo gestionará el acceso a la aplicación. Cada vez que un profesor introduzca sus credenciales, se conectará con la base de datos para verificar que existe una entrada en la tabla de profesores con esas credenciales. Si es así se le permitirá el acceso a la aplicación. Si no es así se le notificará que no se encuentra dado de alta en el sistema.

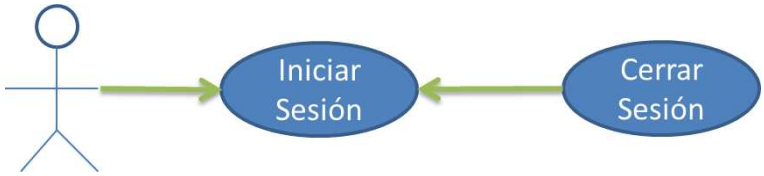

Figura 4.5: Uso módulo de login

- Módulo de gestión de base de datos: Este módulo gestionará los datos de la base de datos. Un profesor podrá crear asignaturas para asignarlas a procesos, podrá crear y modificar profesores y podrá crear y modificar alumnos, así como, asignarle las asignaturas que le correspondan a cada alumno.

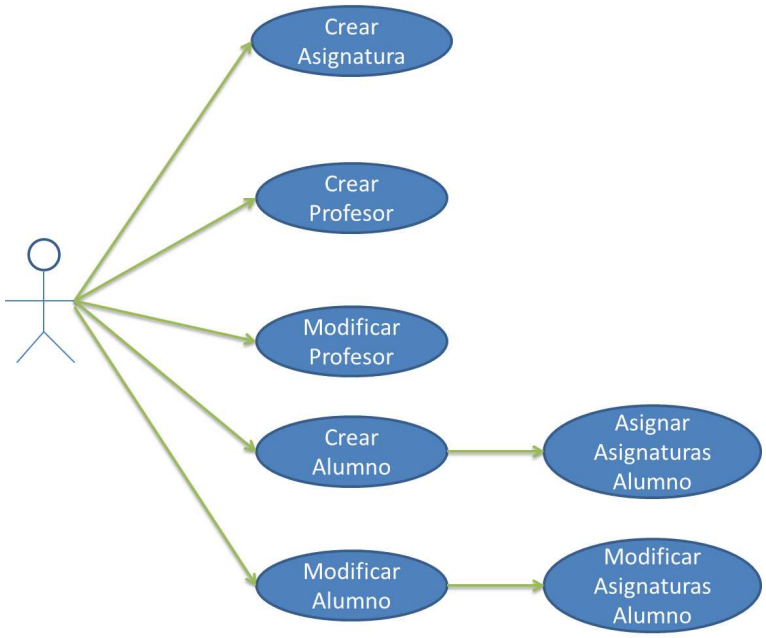

Figura 4.6: Uso módulo de gestión de base de datos

- Módulo de creación de procesos: Este módulo permitirá al profesor crear procesos. Cada vez que seleccione la opción de crear un proceso, se abrirá un módulo para la asignación de datos al proceso. Estos datos serán la asignatura para la que se crea el proceso, el código del proceso y la respuesta correcta del proceso. Finalmente el profesor podrá guardar el proceso una vez asignados todos los datos correspondientes.

Cuando el proceso es creado se almacenará en la base de datos para su posterior tratamiento.

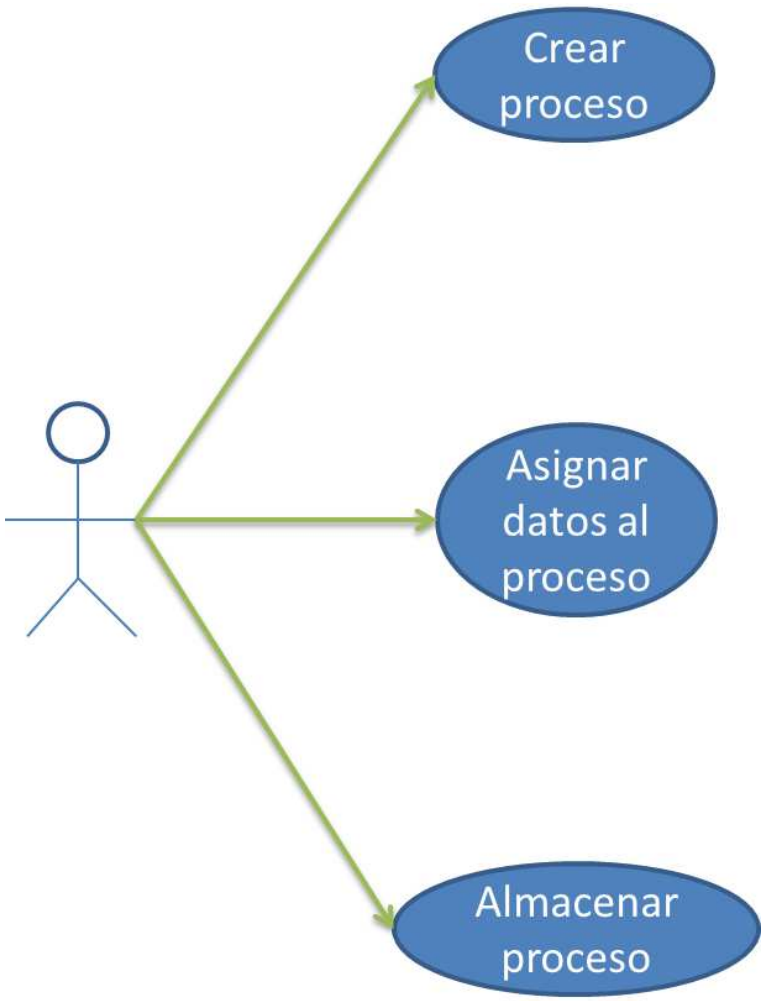

Figura 4.7: Uso módulo de creación de procesos

- Módulo de lanzamiento de procesos: Este módulo permitirá al profesor seleccionar un proceso guardado anteriormente y lanzarlo para que los alumnos asociados a la asignatura a la que ha sido asignado el proceso puedan votar la respuesta que consideren correcta.
  Tras seleccionar la opción de lanzar proceso, se abrirá una pantalla donde el profesor podrá visualizar los procesos que ha creado y desde donde podrá lanzar el proceso que considere oportuno para que los alumnos realicen su voto.

Cuando el proceso sea lanzado se accederá a la base de datos para guardar el proceso como lanzado.

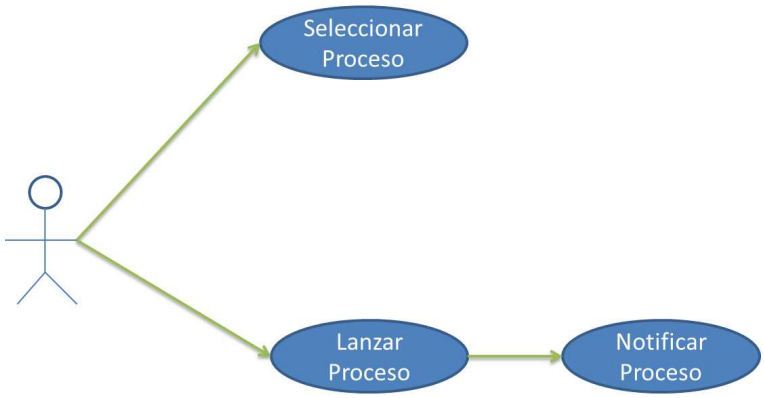

Figura 4.8: Uso módulo de lanzamiento de procesos

- Módulo de visualización de procesos: Este módulo permitirá al profesor seleccionar un proceso guardado y lanzado anteriormente y visualizar la respuesta que ha dado cada alumno, así como, visualizar un gráfico estadístico con la cantidad de votos para cada una de las opciones.
  Para traer toda esta información a la web la aplicación accederá a la base de datos de donde extraerá toda la información de cada proceso.

Figura 4.9: Uso módulo de visualización de procesos

- Módulo de interfaz gráfica: Este módulo se compondrá de cada una de las interfaces de usuario que interactuarán con el profesor a la hora de ejecutar cada una de las opciones anteriormente descritas.

4.3.3. Aplicación del Alumno

Esta aplicación se implementará para dispositivos móviles que cuenten con el sistema operativo Android. Debido a que es necesario enviar información a la base de datos desarrollada en el servidor, será necesaria una conexión a Internet para poder establecer las conexiones necesarias. En este apartado se diseñará la aplicación haciendo especial hincapié en las conexiones que se han de establecer entre el móvil y el servidor.

Diseño arquitectónico

En esta aplicación interactuarán distintos sistemas. De este modo, el diagrama de la arquitectura quedará del siguiente modo:

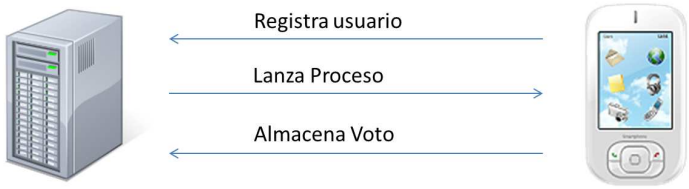

Figura 4.10: Diseño arquitectónico

A continuación se mostrarán cada uno de los componentes que será necesario implementar en la aplicación para cumplir con cada una de las necesidades vistas en el análisis. Para ello se mostrará un diagrama de cada uno de los componentes y sus relaciones entre sí, para después, definir cada uno de ellos y especificar la necesidad que cubre.

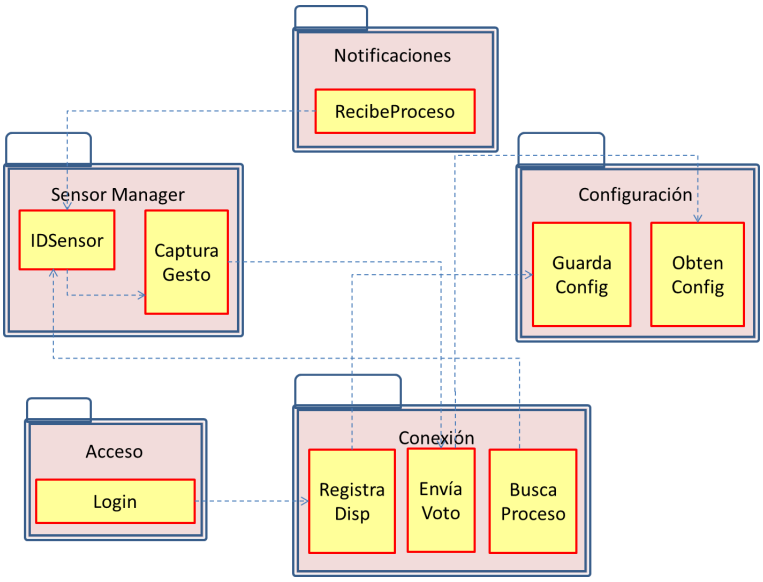

Figura 4.11: Componentes de la aplicación Android

Por cada uno de los componentes anteriores se va a detallar la especificación de todos ellos aportando la siguiente información:

- Descripción: Se dará un detalle descriptivo de cada componente.
- Propósito: Se indicará el propósito que cumplirá cada componente dentro de la aplicación.
- Función: Se dará un listado del uso de cada componente en la aplicación y se describirán cada una de las funciones que realizan.
- Referencias: Se especificará la necesidad vista en el análisis por la que surge este componente.

Cada uno de los componentes que será necesario desarrollar son los siguientes:

- Componente Login: Este componente mostrará una ventana para que el usuario introduzca sus credenciales:
  - Proposito: El propósito de este componente es permitir al usuario introducir sus credenciales en el dispositivo con el que realizará los votos al servidor.

- Función: Recoger las credenciales (usuario y contraseña) que el usuario ha introducido
- Referencias: Este componente hace referencia al requisito de usuario ALUM_1.

◦ Componente ConfigServ: Este componente mostrará una pantalla en la que se introducirá la dirección IP del servidor:

- Proposito: El propósito de este componente es recoger la IP introducida en la pantallas por el usuario.
- Funcion: Guardará la IP del servidor como un parámetro de la configuración.
- Referencias: Este componente hace referencia al requisito de usuario ALUM_2.

◦ Componente RegistraDisp: Este componente se conectara con el servidor para registrar el dispositivo en el que se ha validado el usuario:

- Propósito: El propósito de este componente es obtener un identificador dentro de la aplicación después de que el usuario introduzca sus credenciales.
- Función: Obtendrá un identificador de la aplicación con las credenciales del usuario, lo almacenará en la base de datos del servidor y en el dispositivo móvil.
- Referencias: Este componente hace referencia al requisito de usuario ALUM_1.

◦ Componente GuardaConfig: Este componente almacenará en el dispositivo móvil, tanto, la IP del servidor introducida como el usuario y la contraseña del usuario para acceder a la aplicación directamente una vez que han accedido por primera vez a ella:

- Propósito: El propósito de este componente es almacenar la información en el dispositivo de cada usuario relevante para el acceso a la aplicación.
- Función: Guardar en un fichero XML toda la información de acceso a la aplicación.
- Referencias: Este componente hace referencia a los requisitos de usuario ALUM_1, ALUM_6.

- Componente ObtenConfig: Este componente leerá la información de usuario almacenada en el dispositivo móvil.
  - Propósito: El propósito de este componente es leer las credenciales y la dirección IP del servidor.
  - Función: Si las credenciales y la IP del servidor existen y son correctas, permitirá el acceso inmediato a la aplicación, sino, permitirá al usuario registrarse en la aplicación.
  - Referencias: Este componente hace referencia a los requisitos de usuario ALUM_1, ALUM_6.
- Componente BuscaProceso: Este componente buscará si existe algún proceso disponible para que el usuario registrado realice el voto.
  - Propósito: El propósito de este componente es buscar en el servidor si existe algún proceso que requiera el voto del alumno registrado en el dispositivo.
  - Función: Se conectará con el servidor y buscará en la base de datos los procesos relacionados con el usuario registrado en el dispositivo.
  - Referencias: Este componente hace referencia al requisito de usuario ALUM_7.
- Componente RecibeProceso: Este componente mostrará una notificación en el dispositivo móvil cada vez que el profesor lance un proceso de voto.
  - Propósito: El propósito de este componente es notificar al alumno registrado en el dispositivo de que se ha lanzado un proceso que requiere su voto.
  - Función: La función que realizará este componente será la de mostrar la notificación en el dispositivo.
  - Referencias: Este componente hace referencia al requisito de usuario ALUM_3.
- Componente IDSensor: Este componente detectará un cambio en los sensores que describirán el movimiento para enviar el voto.
  - Propósito: El propósito de este componente es identificar que sensor se ha ejecutado.

- Función: Identificar la ejecución de los sensores que capturen el movimiento.
- Referencias: Este componente hace referencia al requisito de usuario ALUM_4.

◦ Componente CapturaGesto: Este componente realizará los cálculos necesarios para distinguir cual ha sido el movimiento realizado por el alumno.

- Propósito: El propósito de este componente es detectar el cambio en los sensores para identificar la dirección del movimiento.
- Función: Dependiendo del cambio de los sensores, este componente, reconocerá si el movimiento ha sido arriba, abajo, izquierda o derecha.
- Referencias: Este componente hace referencia al requisito de usuario ALUM_4.

◦ Componente EnviaVoto: Este componente enviará el voto capturado por el gesto al servidor.

- Propósito: El propósito de este componente es enviar el voto al servidor, mostrarle al alumno el voto realizado y permitir la correción del voto seleccionado antes de enviarlo.
- Función: Conectarse con el servidor para almacenar el voto realizado por el alumno y mostrar una ventana indicando el voto que se ha realizado permitiendo la correción de este mediante un botón.
- Referencias: Este componente hace referencia a los requisitos de usuario ALUM_4, ALUM_5 y ALUM_8.

Guardado de datos

Los dispositivos móviles Android disponen de una base de datos interna SQLite, pero debido a que la cantidad de datos que se han de almacenar para el correcto funcionamiento de la aplicación es muy pequeña no utilizaremos la base de datos como tal. En su lugar se utilizará la herramienta proporcionada por el API de Android denominada SharedPreferences.
De hecho, este componente, esta específicamente diseñado para almacenar los datos de configuración del usuario dentro de la aplicación. A diferencia de la base de datos que almacena la información en ficheros binarios, este método, los almacena en ficheros XML. El formato que tendrá el fichero de

preferencias de esta aplicación es el siguiente:

```
1  <?xml version='1.0' encoding='utf-8' standalone='yes' ?>
2  <map>
3      <string name="usuario">Ismael</string>
4      <string name="clave">xxxxxx</string>
5      <string name="IPServidor">192.168.1.12</string>
6  </map>
```

Figura 4.12: XML SharedPreferences

Los parámetros del fichero XML son los siguientes:

- Usuario: Es el NIA que ha introducido el usuario a la hora de registrarse en la aplicación. Será para este usuario, para el que se reciban los procesos de voto en los que participe y el usuario con el que se almacenará en la base de datos cada vez que realice un voto.
- Clave: Es la clave con la que se registro el usuario. Solamente es necesaria para verificar que el usuario con el que se intenta dar de alta en la aplicación es correcto.
- IPServidor: Es la dirección IP del servidor de la aplicación.

Diseño de la interfaz de usuario

En este apartado se mostrarán cada una de las interfaces de usuario diseñadas para la aplicación, viendo al mismo tiempo la manera en la que el alumno podrá interactuar con ellas y el resultado de cada una de sus acciones.

Nada más arrancar la aplicación el alumno se encontrará con la siguiente pantalla:

Figura 4.13: Interfaz Gráfica 1

En ella el alumno introducirá la IP y el puerto del servidor. Si esta es correcta y existe conectividad se mostrará la siguiente interfaz en la que el usuario introducirá sus datos de registro en la aplicación (NIA y clave):

Figura 4.14: Interfaz Gráfica 2

Posteriormente aparecerá la siguiente pantalla en la que solo existirá un botón que tras ser pulsado analizará si existen procesos de voto disponibles para el usuario registrado:

Figura 4.15: Interfaz Gráfica 3

Si existe alguno se abrirá esta pantalla en la que el usuario deberá mover el dispositivo arriba, abajo, a la izquierda o a la derecha dependiendo del voto que quiera enviar (a esta pantalla se accederá también desde la notificación recibida cuando el profesor lance un proceso):

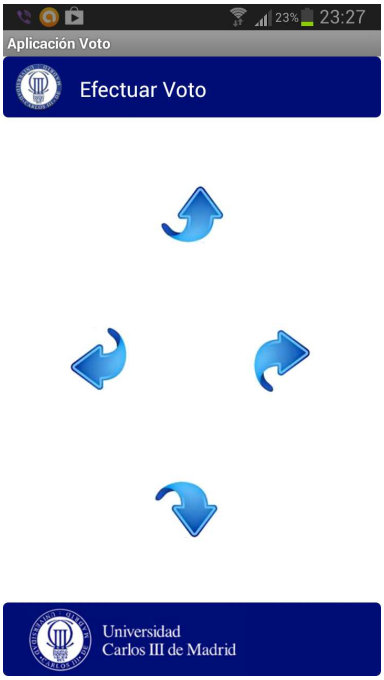

Figura 4.16: Interfaz Gráfica 4

Tras votar aparecerá una ventana de progreso en la que se indicará el voto que se ha realizado:

Figura 4.17: Interfaz Gráfica 5

4.3.4. Integración de las aplicaciones

En este apartado se explicará el flujo que se seguirá a la hora de lanzar un proceso de voto por parte del profesor y que este sea recibido por cada alumno para que envíen los votos mediante el dispositivo móvil, para ser estos, analizados posteriormente por parte del profesor en la aplicación web. En el siguiente esquema se puede comprobar el flujo de la aplicación:

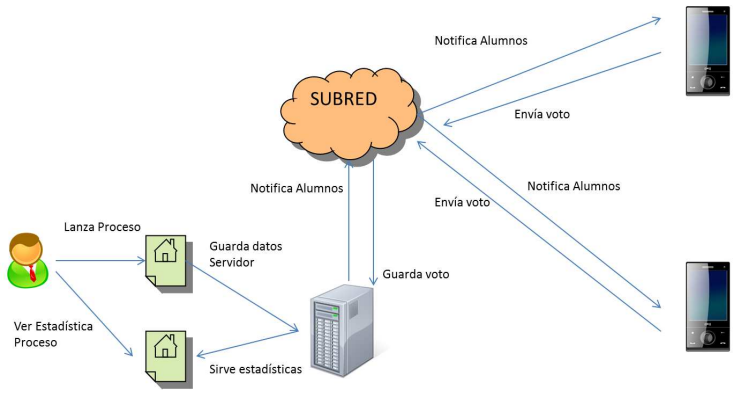

Figura 4.18: Integración de aplicaciones

Como se puede ver en la imagen anterior cada vez que el profesor lance un proceso se notificará a cada uno de los usuarios que sean partícipes en el proceso. Tras esto, cada uno de ellos enviará su voto y se almacenará en el servidor para que el profesor consulte sus estadísticas cuando estime oportuno.

4.4. Implementación

El objetivo de esta etapa es desarrollar la implementación de una aplicación software pueda ser ejecutada correctamente cumpliendo con los criterios especificados en las fases de análisis y diseño. La dificultad reside en cómo realizar esta implementación de la mejor manera posible, ya que va a depender de factores como el lenguaje de programación, la metodología empleada y el entorno de desarrollo establecido.
En este apartado se describirán los aspectos más relevantes del proceso de implementación, como el entorno de desarrollo utilizado, la estructuración del código fuente y la lógica de la aplicación.

4.4.1. Implementación de la base de datos

La base de datos diseñada anteriormente se ha creado en el entorno MySQL. Para ello ha sido necesario instalar el servicio MySQL y generar la base de datos mediante la sentencia CREATE DATABASE PFC;

Para la creación de las tablas se ha utilizado la siguiente sentencia:

CREATE TABLE 'NOMBRE_TABLA' ('CAMPO1' TYPE, PRIMARY KEY ('CAMPO1'), FOREIGN KEY ('CAMPO1') REFERENCES 'TABLA_REF' ('CAMPO_REF'));

Para insertar los datos en las tablas de la base de datos se utilizará la siguiente sentencia:

INSERT INTO 'TABLA' VALUES ('VALOR_CAMPO1', 'VALOR_CAMPO2'...);

Para crear el evento DESACTIVA se lanzará la siguiente sentencia:

CREATE EVENT DESACTIVA ON SCHEDULE EVERY 1 SECOND COMMENT 'Desactivar proceso para enviar voto' DO UPDATE procesos SET activo = '' WHERE activo = 'X' and lanzado = 'X' and hora_fin menor o igual que CURTIME();

4.4.2. Implementación de la aplicación del profesor

Estructura del código

El código se estructurará según la parametrización que propone por defecto el programa eclipse a la hora de la creación de un proyecto web dinamico. Dicha estructura es la siguiente:

- Carpeta WebContent
  - Ficheros jsp: Contendrán las interfaces graficas de la aplicación.
  - Ficheros css: Contendrán los estilos que se aplicarán en cada una de las interfaces gráficas.
  - Carpeta WEB-INF

- Fichero web.xml: Será el fichero descriptor de la aplicación web.
- Carpeta lib: contendrá cada uno de los ficheros jar necesarios para el desarrollo de la web.

- Carpeta images: Carpeta que contendrá cada una de las imágenes necesarias para la aplicación web.

- Carpeta Build: Esta carpeta contendrá cada uno de los ficheros binarios classes que se formarán cuando se compile la aplicación web de cada una de las clases java implementadas.

- Carpeta SRC: Contendrá la estructura de paquetes y las clases java desarrolladas.

Conexión con la base de datos

Para establecer la conexión con la base de datos se ha utilizado un tipo de conector DataSource. Para implementarlo es necesario disponer de un servidor de aplicaciones, en nuestro caso, se utilizará el servidor de aplicaciones de oracle GlassFish.

En dicho servidor es necesaria la configuración de la base de datos. Para ello se añadirá el fichero jar que contiene el conector JDBC de MySQL en la carpeta lib de dicho servidor.

Además es necesaria la configuración del servidor en su consola de administración de la siguiente manera:

- Creación de un Pool de conexiones:

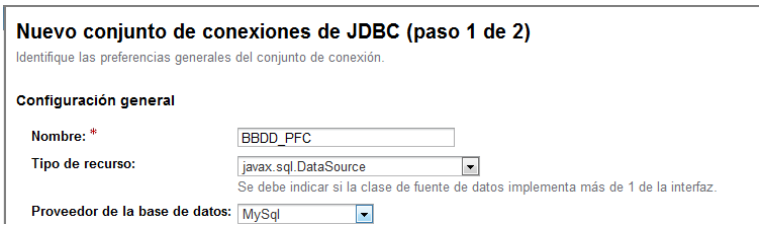

Figura 4.19: Creación Pool de conexiones 1

Se le asignarán las propiedades de la base de datos a este nuevo Pool de conexiones:

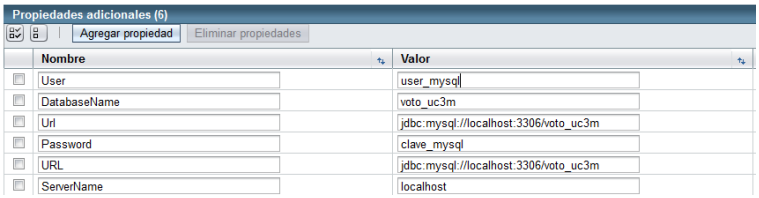

Figura 4.20: Creación Pool de conexiones 2

- Creación del recurso JDBC:

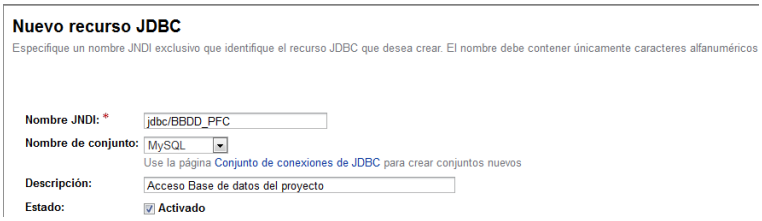

Figura 4.21: Creación Recurso JDBC

- Conexión desde la aplicación web: Para el control de la persistencia con la base de datos desde la aplicación web se ha utilizado el API JPA EclipseLink. Para establecer la conexión con la base de datos desde la aplicación web, dicho API utiliza un fichero XML llamado persistence.xml donde se le indica la clase Java que establece la conexión, así como, cada una de las clases Java que moldearán las tablas de la base de datos. Un ejemplo del fichero es el siguiente:

```
<?xml version="1.0" encoding="UTF-8" ?>
<persistence xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/persistence http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd"
    version="2.0" xmlns="http://java.sun.com/xml/ns/persistence">
    <persistence-unit name="VOTO_UC3M" transaction-type="RESOURCE_LOCAL">
        <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>
        <non-jta-data-source>jdbc/bbdd_pfc</non-jta-data-source>
        <class>Entities.AlumnosByProc</class>
        <class>Entities.Proceso</class>
        <class>Entities.Respuesta</class>
        <class>Entities.Device</class>
        <class>Entities.Usuario</class>


        <properties>
            <property name="eclipselink.ddl-generation" value="create-tables"/>
            <property name="eclipselink.session.customizer" value="ControlBD.Conexion"/>
            <property name="eclipselink.target-database" value="MySQL4"/>
            <property name="eclipselink.logging.level" value="FINEST"/>
        </properties>

    </persistence-unit>
</persistence>
```

Figura 4.22: Fichero persistence.xml

En la captura anterior se puede identificar como se hace referencia al recurso JDBC creado en el servidor en la etiqueta «non-jta-data-source». También se pueden identificar cada una de las clases que modelan la base de datos en las etiquetas «class» y la clase que establece la conexión con la base de datos "ControlBD.Conexion".

Estructura de paquetes y clases

La aplicación está dividida en un alto número de clases. Para estructurarlas y evitar conflictos de nombres entre ellas se han estructurado en paquetes. Cada uno de estos paquetes comprende clases que realizan funciones similares quedando de la siguiente manera:

- ControlBD: Gestiona la conexión con la base de datos y todos los accesos a esta.
- Entities: Contiene cada una de las clases que modelan las tablas base de datos.
- Servlets: Contiene cada uno de los Servlets que son necesarios para el procesado de datos desde la aplicación web.
- GCM: Contiene las clases que implementarán el sistema de notificaciones hacia los dispositivos móviles a la hora de lanzar un proceso por parte del profesor.

A continuación se mostrará una definición de cada paquete y se mostrarán cada una de sus clases:

- ControlBD:
  Este paquete contienete la clase "Conexion" que se encarga de establecer la conexión con la base de datos y cada una de las clases que se encargan de guardar datos en la base de datos o extraerlos. El diagrama de este paquete es el siguiente:

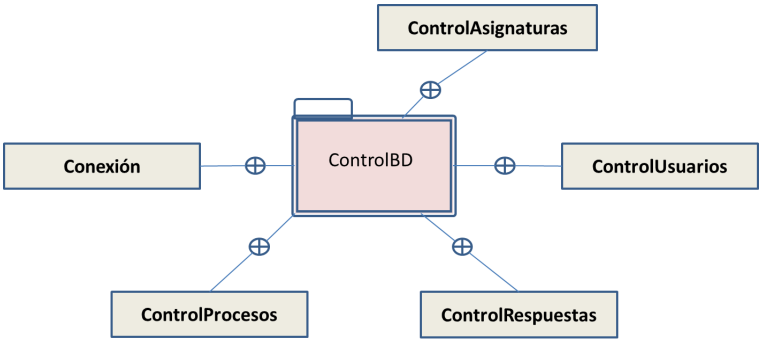

Figura 4.23: Paquete ControlBD

- Entities:

  Este paquete contiene las clases que modelan las tablas de la base de datos. Estas clases implementaran la interfaz Serializable. Las clases cuyo nombre finaliza en PK, modelan la clave primaria de la tabla a la que hacen referencia. El diagrama del paquete es el siguiente:

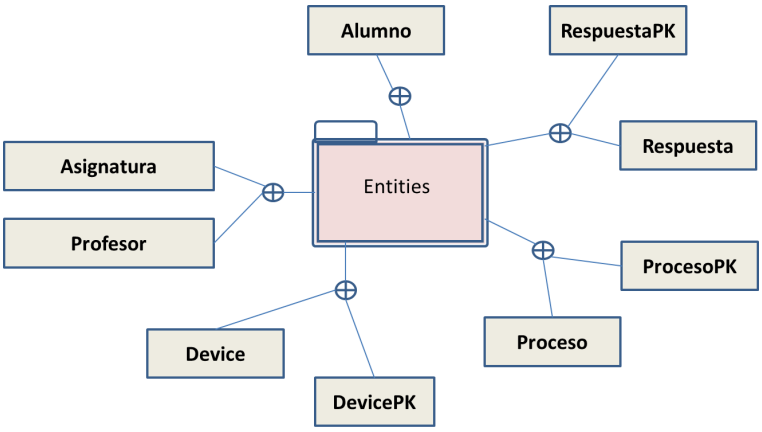

Figura 4.24: Paquete Entities

- Servlets:

  Este paquete contiene las clases que harán el procesado de datos de la aplicación web cuando el profesor utilice la aplicación para realizar alguna acción. Todas estas clases heredan de la clase HttpServlet. El diagram de clases del paquete es el siguiente:

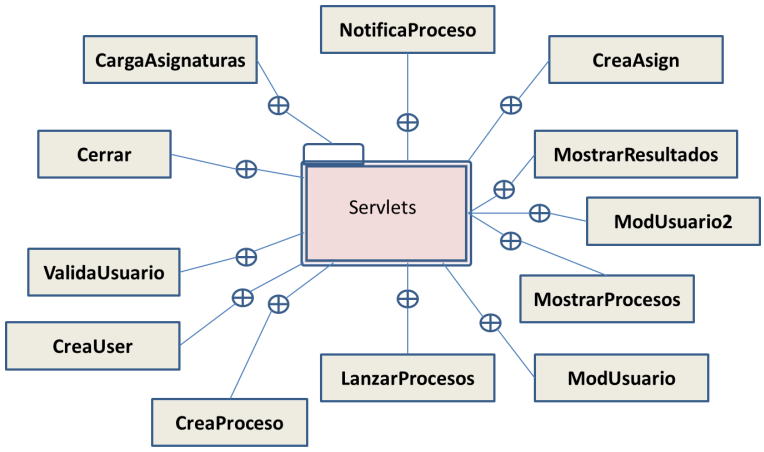

Figura 4.25: Paquete Servlets

- GCM:

  Este paquete contendrá las clases que enviarán la notificación a los dispositivos móviles cuando el profesor lance un proceso. El diagrama del paquete es el siguiente:

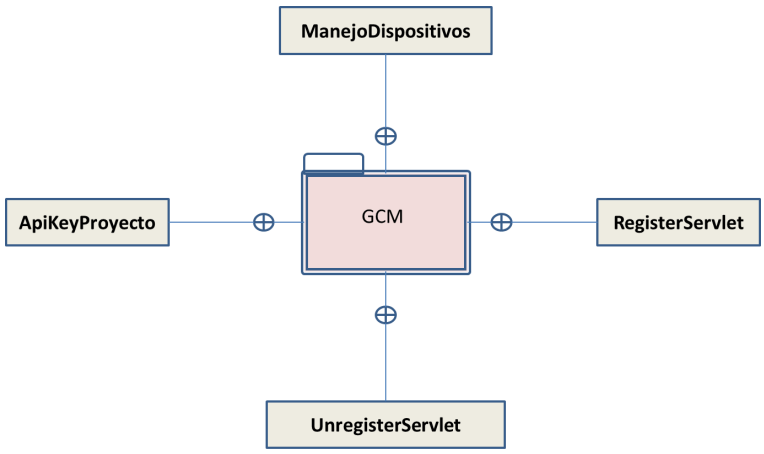

Figura 4.26: Paquete GCM

Acceso a la aplicación

Se mostrará un formulario HTML 5.0 donde el profesor introducirá su NIA y su clave. Al pulsar el botón "Acceder", dicho formulario se conectará con la base de datos para comprobar que el usuario y la contraseña introducidos

corresponden con un profesor en la base de datos.

Gestión de la base de datos

Este componente dispondrá de tres opciones a las que se podrá acceder desde el menu de navegación. Las tres opciones serán las siguientes:

- Creación de asignaturas: El profesor podrá crear las asignaturas necesarias. Para ello, solamente, tendrá que rellenar el nombre de la asignatura en el formualrio que se le mostrará que contendra un «input type = “text”».
- Creación de alumnos: El profesor podrá dar de alta los alumnos que participen en el proceso de voto y asignarlos a las asignaturas que considere oportunas. Dichas asignaturas, se mostrarán en una «table» que contendrá una columna con el nombre de la asignatura y otra con un «input type=“checkbox”» para seleccionar las asignaturas.
- Modificación de usuario: El profesor podrá modificar la contraseña de otros profesores, alumnos o modificar la asignación de asignaturas a un alumno. El profesor accederá al usuario via nia mediante un «input type=“text”» y tipo de usuario (alumno o profesor) mediante un grupo de radiobuttons «input type=“radio”» y se le mostrará otro formulario para modificar los parametros del usuario elegido. Si es un profesor tan solo podrá modificar la contraseña, mientras que, si es un alumno, podrá modificar la contraseña y la asignación de asignaturas de este que se mostrarán en tablas.

Creación de procesos

El profesor dispondrá para crear procesos de un formulario HTML 5.0 donde introducirá los parámetros para la creación de este. Este formulario quedará del siguiente modo:

```
<form id='creaProc' method='post' action='CreaProceso'>
    <h3>Crear proceso de voto:</h3>
    <fieldset>
        <legend>Datos del Proceso</legend>

        <ul>
            <li>
                <label for='code'>Código:</label>
                <input type='text' name='code' id='code' placeholder='Código Proceso' required='required'
            </li>
        </ul>

        <ul>
            <li>
                <label for='Respuesta'>Respuesta:</label>
                <select name="Respuesta">
                    <option value="Arriba">Arriba</option>
                    <option value="Abajo">Abajo</option>
                    <option value="Izquierda">Izquierda</option>
                    <option selected="selected" value="Derecha">Derecha</option>
                </select>
            </li>
        </ul>

    </fieldset>
    <fieldset>
        <legend>Asignatura</legend>

        <ul>
            <li>
                <label for='asign'>Asignatura:</label>
                <select name='asign' id='asign' required='required' >
                    <jsp:scriptlet><![CDATA[ Vector<String> asign = ControlAsignaturas.getAsignaturas();
                    for(int i = 0; i<asign.size();i++){     ]]></jsp:scriptlet>

                    <OPTION> <jsp:expression> asign.get(i) </jsp:expression> </OPTION>

                   <jsp:scriptlet>}</jsp:scriptlet>
                </select>
            </li>
        </ul>
    </fieldset>

    <input type="submit" name="crear" value="Crear" />
</form>
```

Figura 4.27: Creación de procesos

Mediante el “select options” de asignaturas podrá seleccionar la asignatura a la que asociara el proceso a crear.

Lanzamiento de procesos

Si el profesor selecciona la opción de lanzar procesos, se mostrarán cada uno de los procesos generados en el apartado anterior que no hayan sido lanzados hasta el momento. Se mostrarán en un formulario que contendrá una tabla HTML en la que el profesor seleccionará la fila en la que se encuentre el proceso a lanzar mediante un RadioButton situado en un campo de la tabla. La tabla se creará del siguiente modo:

```
59          <form method='get' action="NotificaProceso">
60              <table class="tablaProc" summary="Procesos" >
61                  <jsp:scriptlet>
62                  int filas = tabla.length;
63                  int columnas = tabla[0].length;
64                  </jsp:scriptlet>
65                  <thead>
66                  <tr>
67                      <jsp:scriptlet><![CDATA[ for (int x=0; x<columnas+1; x++){ ]]></jsp:scriptlet>
68                          <th class="filaNombres"><jsp:expression> colNames[x] </jsp:expression></th>
69                      <jsp:scriptlet>}</jsp:scriptlet>
70
71                  </tr>
72                  </thead>
73
74                  <tbody>
75                  <jsp:scriptlet><![CDATA[ for (int i=0; i<filas; i++){ ]]></jsp:scriptlet>
76                      <jsp:scriptlet>String id = tabla[i][0];
77                              pageContext.setAttribute("id", id);
78
79                      </jsp:scriptlet>
80
81                      <tr>
82                          <jsp:scriptlet><![CDATA[ for (int n=0; n < columnas; n++){ ]]></jsp:scriptlet>
83                              <td class="filaNombres"><font color="blue"><jsp:expression> tabla[i][n] </jsp:expression></font></td>
84
85                      <jsp:scriptlet> } </jsp:scriptlet>
86
87                      <td class="filaNombres" align="center"><input type="radio" name="lanzado" value="${ id }" /> </td>
88
89                      </tr>
90
91                  <jsp:scriptlet> }</jsp:scriptlet>
92                  </tbody>
93              </table>
94              <input id="Lanzador" type="submit" name="lanzar" value="Lanzar Proceso" onClick="apagar()"/>
95      </form>
```

Figura 4.28: Lanzamiento de procesos

Como se puede comprobar, tras pulsar el botón “Lanzar Proceso” se enviarán los datos al Servlet “NotificaProceso” que se encargará de llamar al método “lanzarProceso” de la clase “ControlProcesos” para marcar el proceso en la base de datos como lanzado y activo, además, de obtener los dispositivos a los que se ha de notificar y enviarles el mensaje de notificación mediante el mecanismo GCM (Google Cloud Messaging). De esta manera se obtendrán los dispositivos de los alumnos que han de participar en el proyecto y se enviará la notificación a través de este mecanismo.

Visualización de procesos

Si el profesor selecciona esta opción se mostrarán todos los procesos que han sido lanzados y que no están activos mediante un formulario en el que se mostrará una tabla HTML como la del apartado anterior. En esta tabla, el profesor seleccionará la fila del proceso que desee visualizar mediante una columna que contendrá un radiobutton. Tras ello, se accederá otra pantalla donde se visualizará una tabla HTML con la relación de alumnos que han votado en el proyecto así como, un gráfico con los resultados estadísticos de la votación.

Dicho gráfico se generará mediante el API de Google Chart del siguiente modo:

```
<script type="text/javascript" src="https://www.google.com/jsapi"></script>
<script type="text/javascript">
  google.load("visualization", "1", {packages:["corechart"]});
  google.setOnLoadCallback(drawChart);
  function drawChart() {
     var derecha = <%=dere%>;
    var izquierda = <%=izq%>;
    var arriba = <%=up%>;
    var abajo = <%=down%>;

    var data = google.visualization.arrayToDataTable([
      ['Respuestas', 'Votos por proceso'],
      ['Derecha',     derecha],
      ['Izquierda',    izquierda],
      ['Arriba',    arriba],
      ['Abajo',     abajo],
      ['En blanco', 3]
    ]);

    var options = {
      title: 'Respuestas Votadas'
    };

    var chart = new google.visualization.PieChart(document.getElementById('chart_div'));
    chart.draw(data, options);
  }
</script>
```

Figura 4.29: Visualización de procesos

Estructura de la interfaz gráfica

La interfaz gráfica se compondrá de un conjunto de JSP (Java Server Pages) en los que se contendrá toda la parte de la vista gráfica de la aplicación para separarla de la parte del procesado y modelado de datos siguiendo el Modelo-Vista-Controlador (MVC).

Estos JSPs se compondrán de lenguaje HTML y javascript. Además estos JSPs se han organizado del siguiente modo:

- Panel de navegación: Aparecerá el menú de navegación de la aplicación.
- Contenido: Se mostrará el contenido de cada JSP (tablas, formularios, etc).

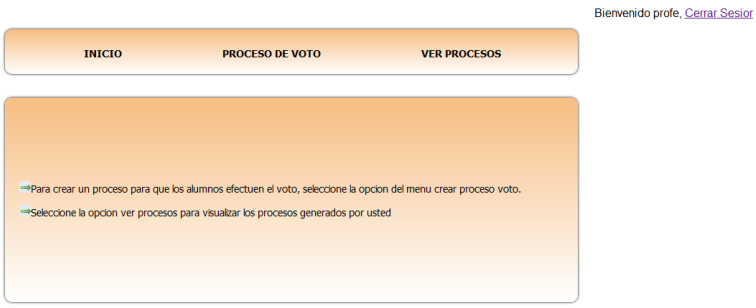

Figura 4.30: Interfaz gráfica

4.4.3. Implementación de la aplicación del alumno

En esta sección se van a comentar los aspectos más importantes de la implementación y que han supuesto un mayor desafío a la hora de desarrollar la aplicación para dispositivos móviles Android.

No es objetivo de esta sección realizar un tutorial de programación sino exponer los conocimientos o experiencias más importantes que se han ido adquiriendo a medida que se ha ido implementando dicha aplicación.

Si se desea visualizar el código fuente relativo a la implementación del proyecto se adjunta en esta memoria.

Conexión con el servidor

La conexión con el servidor se ha realizado por medio de WebServices (Servicios Web) que no son más que un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. De esta manera, diferentes aplicaciones desarrolladas en distintos lenguajes de programación pueden intercambiar datos a través de Internet.

Como hemos comentado anteriormente los servicios web no son más, que un conjunto de protocolos y estándares que en conjunto son capaces de enviar datos entre aplicaciones sobre Internet. Los protocolos utilizados son:

- XML: Se utiliza para la descripción de los datos en el servicio web, es decir, es el modo en que viajan los datos de una aplicación a otra. Por ello, se pueden conectar dos aplicaciones que tengan distintos orígenes y lenguajes de programación.
- WSDL: Se utiliza para la descripción del servicio web. Es un protocolo basado en XML que se puede considerar como el manual de operaciones de los servicios web debido a que nos indica las interfaces que provee este servicio web y los tipos de datos necesarios para su uso.
- SOAP: Es un protocolo estándar que define como dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. SOAP puede funcionar sobre cualquier protocolo de transporte como HTTP, SMTP, TCP... En nuestro caso las peticiones SOAP funcionan a través de HTTP.

Los pasos que se realizan a la hora de consumir un servicio web desde una aplicación, son los siguientes:

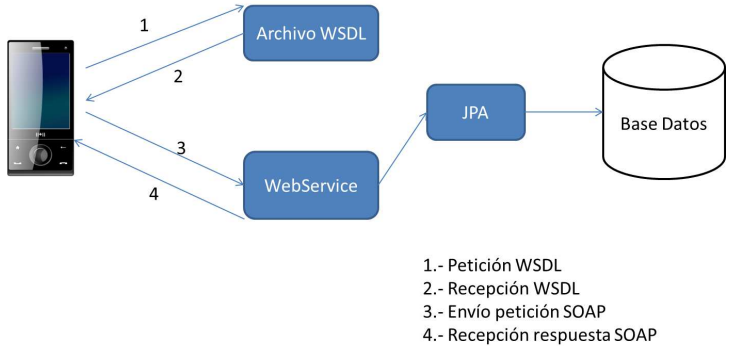

Figura 4.31: Proceso de un WebService

1. El cliente (en este caso el móvil) lo primero que hace es tomar la definición del archivo WSDL.

2. El servidor entrega el fichero WSDL, donde se indica los métodos y propiedades del servicio web que se encuentran disponibles.

3. El cliente hace la petición en función de las especificaciones del fichero WSDL.

4. El servidor entrega el resultado de la consulta.

Se han utilizado conexiones de este tipo tanto para el registro del usuario, el envío de la respuesta al servidor y la búsqueda de procesos de voto disponibles para el usuario registrado en el dispositivo. Para exponer el ejemplo en este apartado utilizaremos el servicio web desarrollado para la validación del usuario a la hora de registrarse en la aplicación.

1. En el servidor se desarrolla el servicio web que es un método en JAVA que realiza la lógica de negocio necesaria para validar el usuario que recibe por parámetro (usuario y clave). La imagen siguiente muestra el método desarrollado:

```
30@    @WebMethod(operationName = "CompUser")
31     public boolean getUsuario(@WebParam(name = "user")String login, @WebParam(name = "pass")String pass){
32         Usuario u = null;
33         boolean user = false;
34
35         EntityManagerFactory factory = Persistence
36                 .createEntityManagerFactory("VOTO_UC3M");
37         EntityManager em = factory.createEntityManager();
38
39         try {
40             Query q = em.createQuery("select u from Usuario u where u.login = '" +login+ "' and u.password = '" +pass+ "'");
41
42             if (q.getResultList() != null) {
43
44                 u = (Usuario)q.getSingleResult();
45                 if (u != null){
46
47                 user = true;
48                 }
49             }
50
51         } catch (NoResultException nrs) {
52             //return 1;
53         } finally {
54             em.close();
55             factory.close();
56         }
57         return user;
58     }
```

Figura 4.32: Metodo WebService

Las etiquetas WebMethod y WebParam indican las especificaciones que han de exponerse en el fichero WSDL.

2. Una vez creado el servicio web y expuesto en el servidor se crea automáticamente el fichero WSDL quedando del siguiente modo:

```
- <definitions targetNamespace="http://WS/" name="WebServicesVotoService">
  - <types>
    - <xsd:schema>
        <xsd:import namespace="http://WS/" schemaLocation="http://192.168.1.12:8888/WebService_UC3M/WebServicesVotoService?xsd=1"/>
      </xsd:schema>
    </types>
  - <message name="CompUser">
      <part name="parameters" element="tns:CompUser"/>
    </message>
  - <message name="CompUserResponse">
      <part name="parameters" element="tns:CompUserResponse"/>
    </message>
  - <portType name="WebServicesVoto">
    - <operation name="CompUser">
        <input wsam:Action="http://WS/WebServicesVoto/CompUserRequest" message="tns:CompUser"/>
        <output wsam:Action="http://WS/WebServicesVoto/CompUserResponse" message="tns:CompUserResponse"/>
      </operation>
    </portType>
  - <binding name="WebServicesVotoPortBinding" type="tns:WebServicesVoto">
      <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    - <operation name="CompUser">
        <soap:operation soapAction=""/>
      - <input>
          <soap:body use="literal"/>
        </input>
      - <output>
          <soap:body use="literal"/>
        </output>
      </operation>
    </binding>
  - <service name="WebServicesVotoService">
    - <port name="WebServicesVotoPort" binding="tns:WebServicesVotoPortBinding">
        <soap:address location="http://192.168.1.12:8888/WebService_UC3M/WebServicesVotoService"/>
      </port>
    </service>
</definitions>
```

Figura 4.33: Fichero WSDL

En la etiqueta schemaLocation se encuentra la URL http:192.168.1.12:8888WebService_UC3MWebServicesVotoService?xsd=1. En esta, se encuentra la definición de cada uno de los parámetros del servicio web expuesto en este ejemplo (usuario y contraseña) como se puede apreciar en la siguiente imagen:

```
- <xs:complexType name="CompUser">
  - <xs:sequence>
      <xs:element name="user" type="xs:string" minOccurs="0"/>
      <xs:element name="pass" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
- <xs:complexType name="CompUserResponse">
  - <xs:sequence>
      <xs:element name="return" type="xs:boolean"/>
    </xs:sequence>
  </xs:complexType>
```

Figura 4.34: Fichero Schema

Se observa que tanto el usuario como el password tienen el formato string y que en la respuesta del servicio web será un booleano (true o false) que indicará si existe el usuario o no.

3. Una vez expuesto el servicio web con su WSDL formado solo queda realizar la llamada desde el dispositivo móvil android. Para ello se utiliza una librería expuesta por Android denominada KSOAP (se ha importado esta librería a la aplicación para dispositivos móviles):

   - Lo primero que se ha de realizar es definir las propiedades del servicio web en cuatro variables de tipo string:

```
final String NAMESPACE = "http://WS/";
final String URL = "http://"+ip+"/WebService_UC3M/WebServicesVotoService?wsdl";
final String METHOD_NAME = "CompUser";
final String SOAP_ACTION = "http://WS/CompUser";
```

Figura 4.35: Propiedades del WebService

Todos estos parámetros se han definido anteriormente a la hora de crear el método en el servidor. El parámetro “ip” se recuperará de las SharedPreferences de la aplicación explicadas en el apartado diseño de esta memoria.

   - Lo segundo es crear la variable con la que se realizará la petición SOAP y añadirle el usuario y la contraseña (recuperados de la interfaz en la que el alumno ha introducido el usuario y la

contraseña a la hora de registrarse):

```
SoapObject request = new SoapObject(NAMESPACE, METHOD_NAME);
request.addProperty("user", user);
request.addProperty("pass", pass);
```

Figura 4.36: Adición de parametros a la llamada del WebService

Se ha de notar que “user” y “pass” son los nombres que tienen los parámetros en el fichero WSDL. Tambien es importante mencionar que el formato de user y pass es de tipo string tal y como se indicaba en la definición del servicio web.

- Después se creará un objeto de serialización y se realizará la llamada al servicio web:

```
SoapSerializationEnvelope envelope = new SoapSerializationEnvelope(
        SoapEnvelope.VER11);
envelope.setOutputSoapObject(request);

HttpTransportSE androidHttpTransport = new HttpTransportSE(
        URL);

androidHttpTransport.call(SOAP_ACTION, envelope);
```

Figura 4.37: Ejecución de la llamada al WebService

- Finalmente se obtendrá la respuesta del servicio web:

```
SoapSerializationEnvelope envelope = new SoapSerializationEnvelope(
        SoapEnvelope.VER11);
envelope.setOutputSoapObject(request);

HttpTransportSE androidHttpTransport = new HttpTransportSE(
        URL);

androidHttpTransport.call(SOAP_ACTION, envelope);
```

Figura 4.38: Obtención de la respuesta del WebService

Como se puede comprobar la respuesta la pasamos a un parámetro de tipo booleano debido a que en el WSDL está definido de

esta manera.

Captura del movimiento

Para realizar el voto y detectar si el movimiento realizado por el alumno es arriba, abajo, izquierda o derecha, ha sido necesario utilizar los sensores de los que disponen los dispositivos android. En concreto se han utilizado los siguientes sensores:

1. Sensor brújula/magnetómetro
2. Acelerometro

En el método onCreate de la actividad en la que se realiza el voto es necesario dar de alta los eventos de ambos sensores para que cada vez que se registre un cambio en uno de los sensores se lance el evento para calcular el movimiento mediante el método getSystemService(SensorService). Los sensores se registrarán en el metodo onResume() de la actividad del siguiente modo:

```
@Override
protected void onResume() {
    super.onResume();
    Sensor gsensor = mSensorManager
            .getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
    Sensor msensor = mSensorManager
            .getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD);
    mSensorManager.registerListener(this, gsensor,
            SensorManager.SENSOR_DELAY_GAME);
    mSensorManager.registerListener(this, msensor,
            SensorManager.SENSOR_DELAY_GAME);
}
```

Figura 4.39: Registro de sensores

Cada vez que se registre un cambio en uno de estos dos sensores se lanzará el método onSensorChanged(SensorEvent event). Cada vez que se ejecute este método se obtendrán los datos de los tres ejes de los sensores a partir de los cuales se obtendrá la matriz de rotación mediante el método proporcionado por el API de Android getRotationMatrix(datosAcelerometro, datosMagnetometro). A partir, de esta matriz se podrá obtener la orientación del dispositivo mediante el método proporcionado por el API de

Android getOrientation(matrizRotacion). Este método nos devolverá un array con los tres ángulos de orientación del dispositivo (yaw, pitch y roll). Con la primera obtención de este array se obtendrá la orientación cardinal del dispositivo mediante el ángulo yaw, ya que este ángulo varía entre -180 grados y 180 grados dependiendo de la orientación cardinal del dispositivo.

En los siguientes cambios del sensor se verificará la variación del ángulo yaw con respecto al inicial para obtener el movimiento a derecha o izquierda del dispositivo, mientras que, se verificará la variación de los ángulos roll y pitch para verificar si el movimiento es arriba o abajo.

Recepción de notificaciones

Para el servicio de notificaciones se ha utilizado el tipo de push de notificaciones de Android. Para ello se ha utilizado la herramienta que provee Google denominada GCM (Google Cloud Messaging). Para utilizarlo ha sido necesario crear un proyecto en Google y obtener un API Key (clave identificadora del proyecto) que usará la aplicación para dispositivos móviles.

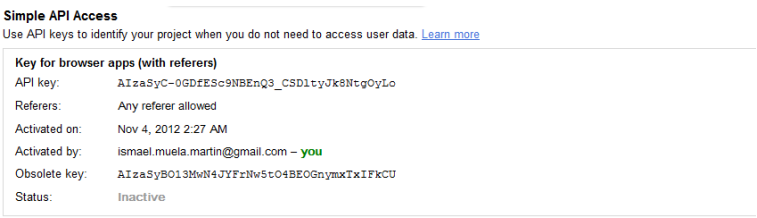

Figura 4.40: Api Key de Google

Para el desarrollo en la aplicación ha sido necesario importar la librería gcm.jar y realizar los siguientes desarrollos:

1. Permisos AndroidManifest.xml: Ha sido necesario dotar a la aplicación para enviar y recibir mensajes del tipo C2D_MESSAGE, de permisos de internet y de permisos para lanzar la aplicación cuando se reciba una notificación:

```
<!-- GCM connects to Google Services. -->
<uses-permission android:name="android.permission.INTERNET" />

<!-- GCM requires a Google account. -->
<uses-permission android:name="android.permission.GET_ACCOUNTS" />

<!-- Keeps the processor from sleeping when a message is received. -->
<uses-permission android:name="android.permission.WAKE_LOCK" />

<permission
    android:name="pfc.ismael.voto.permission.C2D_MESSAGE"
    android:protectionLevel="signature" />

<uses-permission android:name="pfc.ismael.voto.permission.C2D_MESSAGE" />

<!-- This app has permission to register and receive data message. -->
<uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
```

Figura 4.41: Permisos AndroidManifest.xml

2. Crear un receptor de difusión (Broadcast Receiver):

```
<receiver
    android:name="com.google.android.gcm.GCMBroadcastReceiver"
    android:permission="com.google.android.c2dm.permission.SEND" >
    <intent-filter>

        <!-- Receives the actual messages. -->
        <action android:name="com.google.android.c2dm.intent.RECEIVE" />
        <!-- Receives the registration id. -->
        <action android:name="com.google.android.c2dm.intent.REGISTRATION" />

        <category android:name="pfc.ismael.voto" />
    </intent-filter>
</receiver>
```

Figura 4.42: Receptor de difusión

Este será el encargado de recibir las notificaciones y de lanzarla en el dispositivo móvil. Este receptor se encuentra funcionando en todo momento aunque la aplicación se encuentre en segundo plano.

3. Creación de la clase que manejará las acciones del receptor de difusión. Esta clase se ha llamado "GCMIntentService" y hereda de la clase "GCMBaseIntentService". Los métodos a desarrollar en ella son los siguientes:

```
@Override
protected void onRegistered(Context context, String registrationId)
```

Figura 4.43: Metodo onRegistered

- En este método se llamará a un servicio web que registre el dispositivo en la aplicación dada de alta en google y que lo almacena en la base de datos desarrollada para este proyecto en la tabla “Dispositivos”.

```
@Override
protected void onMessage(Context context, Intent intent)
```

Figura 4.44: Metodo onMessage

- Cuando el receptor de difusión ejecute este método se generará la notificación indicando que un nuevo proceso de voto ha sido generado.

Guardado de parametros de la aplicación

Los parámetros que se han guardado en la aplicación son la dirección IP del servidor, con la que se hace el registro. Para ello se ha utilizado el API expuesto por Android llamado SharedPreferences del siguiente modo:

1. Guardado de preferencias:

```
SharedPreferences settings = getSharedPreferences(
        "nia_al", MODE_PRIVATE);
SharedPreferences.Editor editor = settings.edit();
editor.putString("nia_al", un.getText().toString());
editor.commit();
```

Figura 4.45: Guardado de preferencias

2. Obtención de preferencias:

```
SharedPreferences settings2 = getSharedPreferences("ip",
        MODE_PRIVATE);
String ip = settings2.getString("ip", "");
```

Figura 4.46: Obtención de preferencias

3. Modificación de preferencias:

```
SharedPreferences settings = c.getSharedPreferences(
        "ip", MODE_PRIVATE);
 SharedPreferences.Editor editor = settings.edit();
 editor.clear();
 editor.putString("ip", ip.getText().toString());
 editor.commit();
```

Figura 4.47: Modificación de preferencias

Capítulo 5

Pruebas de la aplicación

5.1. Introducción

Las pruebas de la aplicación son muy importantes para verificar el correcto funcionamiento de la misma. De esta manera, se encontrarán los errores que puede contener la aplicación para poder corregirlos conseguir implementar una aplicación de calidad.

De este modo, las pruebas deben ser un proceso destructivo, es decir, no consiste en demostrar que la aplicación no contiene errores, si no, todo lo contrario. Las pruebas se considerarán exitosas si se encuentran los errores de la aplicación.

En el caso de la aplicación que estamos tratando se han realizado dos tipos de prueba:

- Pruebas unitarias: Se han realizado pruebas de cada componente de cada una de las aplicaciones desarrolladas individualmente.
- Pruebas integradas: Se han realizado pruebas en conjunto de cada una de las aplicaciones.
- Pruebas del sistema: Se han realizado pruebas para comprobar el funcionamiento ambas aplicaciones en conjunto.

5.2. Pruebas unitarias

Con estas pruebas se pretende comprobar que cada clase de la aplicación realiza la labor que tiene asignada o para la que fue creada individualmente. Estas pruebas se han ido realizando sobre cada componente según se ha ido desarrollando. Debido a que son pruebas individuales es necesario distinguir en el plan de pruebas entre las dos aplicaciones vistas en esta memoria. De este modo, el plan de pruebas de la aplicación queda del siguiente modo:

- Plan de pruebas de la aplicación web (profesores)

Componente	Descripción	Resultados
Login	Se accederá a la aplicación con un usuario dado de alta	El sistema accede a la aplicación correctamente

Login	Se accederá a la aplicación con un usuario no dado de alta	El sistema no accede a la aplicación y muestra un mensaje indicando que el usuario no esta dado de alta
Creación de asignatura	Se crea una asignatura	Se verifica en la tabla de asignaturas su correcta creación
Creación de alumno	Se crea un alumno y se le asignan asignaturas	Se verifica en las tablas que el alumno ha sido creado y las asignaturas ha sido asignadas
Modificación de alumno	Se modifica la contraseña de un alumno	Se verifica en la base de datos que la contraseña ha cambiado
Modificación de alumno	Se desasigna una asignatura de un alumno	Se comprueba en la base de datos que la asignatura ha sido desasignada
Modificación de alumno	Se asigna una asignatura de un alumno	Se comprueba en la base de datos que la asignatura ha sido asignada
Crear Proceso	Se accederá a la zona para crear un proceso	Se verifica que se accede correctamente a este apartado
Asignar datos a Proceso	Asignación de asignatura	Se comprueba que el proceso ha sido creado para la asignatura seleccionada
Guardar Proceso	Se verificará que el proceso se almacena correctamente en la base de datos	Se verifican las tablas de la base de datos que implican la creación de un proceso para comprobar su correcta creación
Lanzar Proceso	Se accederá a la zona para lanzar procesos creados	Se verifica que accede correctamente a esta zona de la web y que carga los procesos que el profesor tiene creados
Elección de Proceso	Se verificará que cada uno de los procesos que aparecen son seleccionables	Se verifica que aparece un radiobutton al lado de cada proceso para poder seleccionarlo

Lanzamiento de proceso	Se verificará que al pulsar el boton de lanzar proceso modifica el proceso en la base de datos	Se verifica que al pulsar el boton activa el flag en la tabla de procesos que indica que el proceso ha sido lanzado y que posteriormente desaparece de la tabla de procesos disponibles para lanzamiento
Visualizar Proceso	Se accederá a la zona para visualizar procesos lanzados	Se verifica que accede correctamente a esta zona de la web y que carga los procesos que el profesor ha lanzado
Elección de proceso	Se verificará que cada uno de los procesos que aparecen son seleccionables	Se verifica que aparece un radiobutton al lado de cada proceso para poder seleccionarlo
Resultados Proceso	Se verificará que se pueden visualizar los votos que ha recibido el proceso	Se comprueba que aparecen cada uno de los votos recibidos para el proceso y además que se visualiza un gráfico con los votos recibidos

Cuadro 5.1: Plan de pruebas. Aplicación Web

- Plan de pruebas de la aplicación Android (alumnos)

Componente	Descripción	Resultados
Login	Se accederá a la aplicación con un usuario dado de alta	El sistema accede a la aplicación correctamente
Login	Se accederá a la aplicación con un usuario no dado de alta	El sistema no accede a la aplicación y muestra un mensaje indicando que el usuario no esta dado de alta

ConfigServ	Se configurará la IP del servidor donde se encuentra la base de datos	Se verifica que tras poner la IP correcta se accede a la aplicación y que al ponerla incorrecta la vuelve a solicitar
RegistraDisp	Se comprobará que el dispositivo ha sido registrado al acceder a la aplicación	Se verfica en la tabla de dispositivos que se ha registrado correctamente el dispositivo
GuardaConfig	Se accederá por segunda vez a la aplicación	Se verifica que los datos de usuario han sido correctamente almacenados debido a que permite el acceso a la aplicación sin solicitar de nuevo dichos datos
ObtenConfig	Se verificará que se obtienen los datos de usuario almacenados correctamente	Mismo resultado que la prueba anterior
BuscaProceso	Se pulsará el boton votar	Se verifica que aparece un mensaje indicando que no existen procesos activos
RecibeProceso	Ver una notificación	Se ejecuta el metodo que genera notificaciones y se comprueba la correcta recepción de la notificación
IDSensor	Ver datos de los sensores	Se verifica que los sensores están registrados correctamente visualizando los datos de ellos mediante TextView
CapturaGesto	Se realizará el movimiento del telefono	Se verifica que al mover el teléfono en una de las direcciones captura el gesto correcto lanzando una ventana que indica el movimiento realizado

EnvíaVoto	Se verificará que el voto se ha guardado en la base de datos	Se comprueba en la base de datos que la respuesta ha sido almacenada correctamente

Cuadro 5.2: Plan de pruebas. Aplicación Android

5.3. Pruebas integradas

Estas pruebas pretenden la correcta integración de los componentes de cada aplicación. Estas pruebas se han realizado una vez que se ha completado la implementación de todos los componentes de cada una de las aplicaciones. Debido a que es necesario realizar estas pruebas en cada una de las aplicaciones desarrolladas en este proyecto es necesario diferenciar las pruebas en ambas. De este modo, el plan de pruebas queda del siguiente modo:

- Plan de pruebas de la aplicación web (profesores)

  En este caso, se ha realizado una prueba de ciclo completo de la aplicación en la que se han comprobado los siguientes parametros:

  - Al crear un proceso aparece en la tabla que permite lanzar procesos y no en la de visualizar procesos.
  - Al lanzar un proceso, desaparece de la tabla de procesos a lanzar y aparece en la tabla de visualizar procesos.

- Plan de pruebas de la aplicación Android (alumnos) En este caso, se ha realizado una prueba de ciclo completo de la aplicación en la que se han comprobado los siguientes parametros:

  - Se ha comprobado que al realizar el voto de un proceso no nos permite volver a votar de nuevo sobre este.
  - Se ha comprobado que el voto que se envía se asigna correctamente al usuario con el que se ha registrado el dispositivo.

5.4. Pruebas del sistema

Estas pruebas pretenden la correcta integración de todo el sistema desarrollado en este proyecto. Es decir, el correcto funcionamiento de ambas aplicaciones en conjunto. De este modo el plan de pruebas queda del siguiente modo:

- Se crea el proceso y se verifica que se ha almacenado correctamente en la base de datos.
- Se lanza el proceso que se ha creado anteriormente.
- Tras el paso anterior, se verifica que la notificación se ha recibido en el dispositivo móvil.
- Se accede a la aplicación a partir de la notificación y se envía el voto.
- Se visualiza el proceso en la aplicación del profesor y se comprueba que aparece correctamente el voto enviado.

Capítulo 6

Historia del proyecto

6.1. Introducción

Cuando comencé este proyecto lo aborde como un proyecto interesante que me aportaría mucha experiencia y el aprendizaje de lo que era para mí, un universo nuevo para descubrir como es el desarrollo de las aplicaciones para móviles, en concreto, para dispositivos que dispusieran del sistema operativo Android.

La idea propuesta por mi tutor, Mario, consistía en realizar una aplicación que fuera capaz de resolver un ejercicio propuesto por el profesor en clase sin perder tiempo y a través de movimientos del dispositivo y que, además, el profesor pudiera realizar un seguimiento de estos ejercicios.

Sin más, me puse a la tarea, y comencé a investigar como capturar los movimientos del dispositivo móvil y empecé a introducirme en el mundo de los sensores y a implementar el proyecto que se ha explicado en este documento.

6.2. Conclusiones

Para este proyecto se plantearon una serie de objetivos que se han ido cumpliendo a medida que se ha ido desarrollando el proyecto. Los objetivos más importantes que se plantearon y que se han cubierto con este proyecto son:

- La aplicación debía ser realizada para realizar ejercicios en clase sin perder apenas un minuto.
- Eliminar la necesidad de utilizar la mano alzada o de repartir un papel por cada alumno a la hora de resolver los ejercicios.
- Permitir al profesor realizar un seguimiento de cada uno de los ejercicios, así como, ver las estadísticas de respuestas en cada uno de ellos.
- Utilizar el movimiento del dispositivo móvil mediante un gesto manual para resolver el ejercicio.

Cada uno de estos objetivos ha sido cubierto con las dos aplicaciones implementadas y el sistema de información formado con su conjunto.

Gracias a que la aplicación se ha desarrollado para Android, no solo se han cubierto los objetivos, sino que, me ha permitido conocer y aprender esta plataforma y me ha ofrecido la oportunidad de conocer los pasos necesarios a seguir para desarrollar aplicaciones para este sistema.

Desde un punto de vista más personal, la implementación de este proyecto, ha despertado en mí un gran interés por el mundo de los dispositivos móviles y visto como se encuentra el panorama nacional e internacional en este mercado parece que es una buena rama para comenzar.

De este modo, el proyecto, no solo ha cubierto los objetivos por lo que fue planteado inicialmente, sino que, me ha servido para aprender un nuevo tema y para abrir una puerta hacia un futuro profesional.

6.3. Líneas futuras

Tras finalizar el proyecto se han detectado una serie de líneas futuras de implementación que sería oportuno evaluar su realización:

- Desarrollo para otras plataformas móviles: Debido a que todos los alumnos no contarán con un dispositivo móvil con el sistema operativo Android, se podría evaluar el desarrollo para otras plataformas de smartphones.
- Integración con la base de datos de la UC3M: Integrar la parte de los datos de alumnos y profesores con la base de datos de la Universidad, para así, poder notificar a todos los usuarios de la asignatura para la que el profesor cree el ejercicio.
- Introducir otra forma de voto como puede ser el API de Gesture de android con el que dibujando la dirección sobre la pantalla del dispositivo pueda conocer el lado hacía el que se quiere realizar el voto.
- Adaptación de la aplicación web al navegador de los dispositivos móviles.

Capítulo 7

Planificación y presupuesto

7.1. Introducción

En este capítulo se explicará la planificación y el presupuesto del proyecto implementado. Se expondrán detalles sobre la estimación de cada uno de los apartados que componen el desarrollo de una aplicación a medida, así como, el coste del proyecto tanto a nivel de equipos utilizados como de personal.

7.2. Planificación

A continuación, se va a detallar la planificación seguida para la elaboración del proyecto. Se ha de tener en cuenta que la dedicación al proyecto no ha sido exclusiva ni a tiempo completo debido a que se han estado desarrollando otras actividades que no han permitido dedicar todos los esfuerzos al mismo.

A continuación se mostrará una tabla de cada una de las fases de las que ha constado el proyecto y en la que se expondrán la fecha de inicio y de finalización de cada una de estas fases, así como, un total de los días que han durado.

Fase	Fecha de inicio	Fecha de fin	Total días
Estudio previo	01/02/2012	15/03/2012	44 días
Fase de aprendizaje inicial	16/03/2012	31/05/2012	77 días
Analisis	01/06/2012	15/07/2012	45 días
Diseño	16/07/2012	30/09/2012	77 días
Implementación	01/10/2012	01/02/2013	124 días
Pruebas	02/02/2013	16/02/2013	15 días
Memoria	17/02/2013	04/04/2013	47 días

Cuadro 7.1: Planificación del proyecto

En la siguiente figura se muestra el diagrama de Gantt que se encuentra relacionado con la planificación descrita en la tabla anterior:

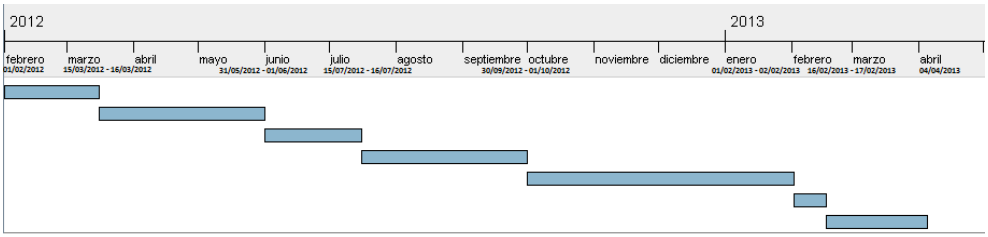

Figura 7.1: Diagrama de Gantt

7.3. Presupuesto

En este apartado de detallará el presupuesto de la implementación de la aplicación teniendo en cuenta el gasto de personal, licencia y equipos siguiendo el modelo propuesto por la Universidad Carlos III de Madrid.[12]

Los gastos de personal son los siguientes:

- Dedicación de 3 horas diarias
- No se cuentan los días festivos:
  - 24, 25 y 31 de Diciembre de 2012
  - 1 y 6 de Enero de 2013
  - 1 de Mayo de 2012
  - 12 de Octubre de 2012

Los días totales de la planificación ascienden a 429 días pero restándole estos 7 días festivos y cuatro días por mes correspondientes a los domingos se quedan en 364 días reales. Con estos datos se pueden obtener los gastos de personal asociado al proyecto con la siguiente fórmula:

Coste de personal = ((Total días x Horas/día) / Dedicación hombre/mes) * Coste hombre/mes
Total días = 364 días
Horas/día = 3
Dedicación hombre mes = 131,25 horas
Coste hombre mes = 2694,39 euros

Por tanto, los gastos de personal se elevan a VEINTIDOS MIL CUATROCIENTOS DIECISIETE CON TREINTA Y DOS CÉNTIMOS.

Para la realización del proyecto, además se ha utilizado un equipo y un dispositivo móvil con los siguientes precios:

- Ordenador portátil ASUS A52J con valor de 599 euros.
- Dispositivo móvil Samsung Galaxy S3 con valor de 450 euros.

Para calcular el valor en el proyecto es necesario calcular la amortización de dichos equipos siguiendo los siguientes parámetros:

Descripción	Coste	%Uso Proyecto	Dedicación (meses)	Periodo Depreciación	Coste Imputable
Portatil	599	100	14	60	139,77
Movil	450	100	14	60	105

Cuadro 7.2: Amortización de equipos

Finalmente se calculan algunos gastos imputables al proyecto:

- Tarifa de red con un coste de 20 euros al mes (14 * 20 = 280 euros)

Finalmente el coste del funcionamiento del proyecto asciende a QUINIENTOS VEINTICUATRO EUROS CON SETENTA Y SIETE CÉNTIMOS.

En resumen el presupuesto final quedaría como se muestra en la tabla siguiente:

Concepto	Importe (Euros)
Personal	22.417,32
Amortización	245
Costes de funcionamiento	280
Costes Indirectos	4.588,46
Total (Sin IVA)	27.530,78
TOTAL (IVA 21 %)	33.312,24

Cuadro 7.3: Amortización de equipos

Finalmente el presupuesto total de este proyecto asciende a TREINTA Y TRES MIL TRESCIENTOS DOCE CON VEINTICUATRO CÉNTIMOS.

Leganés, 15 de mayo de 2013

El ingeniero proyectista

Fdo. Ismael Muela Martín de Bernardo

Apéndice A

Manual de instalación

Debido a que la aplicación de Android solo requiere disponer del paquete APK para poder instalarla en el dispositivo móvil, este manual se centrará, en la instalación de la aplicación web en un equipo que disponga de un servidor de aplicaciones web GlassFish y de la base de datos MySQL requerida por la aplicación.
Desde el IDE de Eclipse se puede comprimir la aplicación web en un fichero WAR que contenga todas las clases necesarias para levantar la aplicación en un servidor. Una vez que tenemos el paquete WAR simplemente se ha de ir al servidor he implementar la aplicación. En GlassFish se realiza la implementación de la aplicación del siguiente modo:
Desde la consola de configuración en el caso de GlassFish se accede mediante la IP http://localhost:4848/, se accederá al apartado de aplicaciones del menú de la izquierda. Posteriormente se pulsará el botón implementar y se cargará el archivo WAR generado anteriormente:

Figura A.1: Implementación de la aplicación en GlassFish

Finalmente pulsaremos el botoón Aceptar y podremos acceder a la aplicación a través de la URL http://localhost:8888/CreaProcesoVotoV3/ Notar que para el correcto funcionamiento de la aplicación también será necesario levantar la aplicación que contendrá los webservices necesarios para la comunicación con el dispositivo móvil. Esta aplicación irá en otro paquete WAR.

Apéndice B

Manual de usuario de la aplicación web

En este apartado se detallará el manejo de usuario de la aplicación web por parte del profesor. De esta forma, se describirán los pasos a seguir para crear un proceso, lanzarlo para que los alumnos emitan su voto y finalmente visualizar las estadísticas de este.

- Acceso a la aplicación: El profesor introducirá su usuario y contraseña y accederá a la aplicación en la que podrá crear procesos de voto:

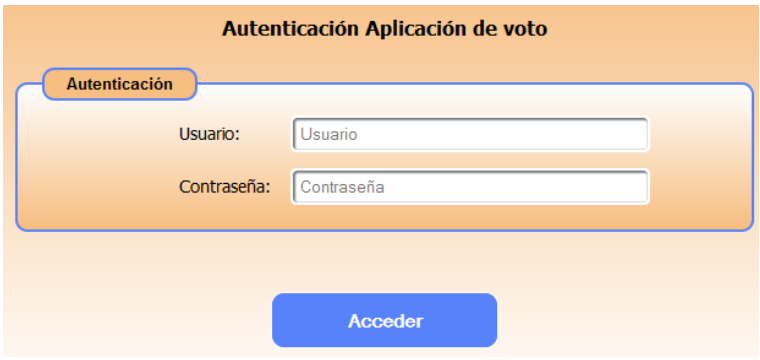

Figura B.1: Acceso a la aplicación

- Creación de asignaturas: Se accederá a través del menú de navegación en la parte superior de la pantalla. Será necesario dotar a la asignatura de un nombre. La pantalla que se visualizaría es la siguiente:

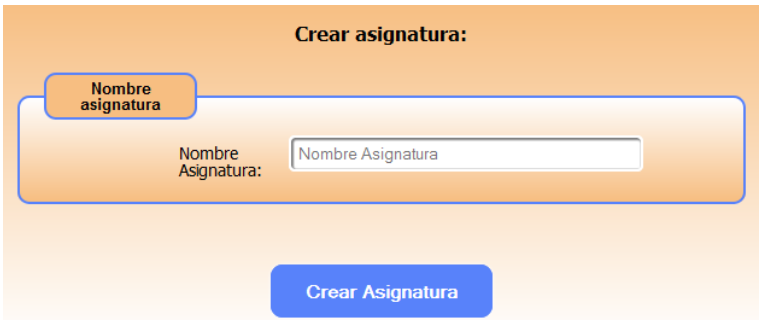

Figura B.2: Creación de asignatura

- Creación de un alumno: Se accederá a través del menú de navegación en la parte superior de la pantalla. Será necesario dotar al alumno de una contraseña y de las asignaturas que va a tener asignadas. La pantalla que se visualizaría es la siguiente:

Figura B.3: Creación de alumnos

- Modificación de usuarios: Se accederá a través del menú de navegación en la parte superior de la pantalla. Se podrán modificar tanto la contraseña para profesor y alumno como las asignaturas a las que se encuentra asignado un alumno. La pantalla que se visulizará serála siguiente:

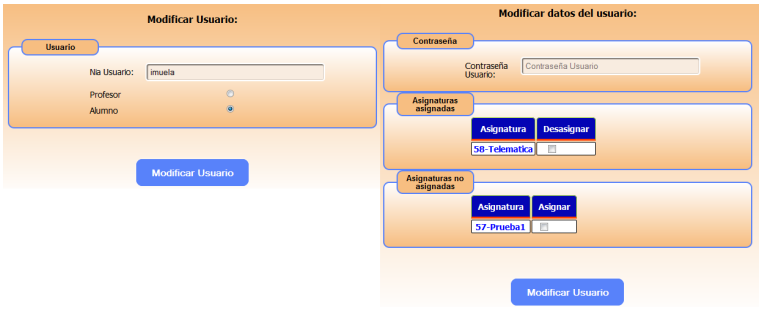

Figura B.4: Creación de alumnos

- Creación del proceso: Se accederá en la web a través del menú de navegación en la parte superior de la pantalla. Será necesario dotar al proceso de los parámetros necesarios para su creación. Estos son:
  - Código: Código identificativo del proceso.
  - Respuesta: Será la respuesta correcta del proceso.
  - Asignatura: Deberá seleccionar la asignatura con la que se quiere relacionar el proceso.

Tras informar cada uno de los datos necesarios, se pulsará el botón "Crear". Si todo va bien la aplicación volverá a mostrar esta pantalla. Si existe algún error a la hora de crear el proceso el usuario será informado de dicho error.
De esta manera, el proceso será guardado, en espera de que el profesor lo lance para que los alumnos emitan el voto:

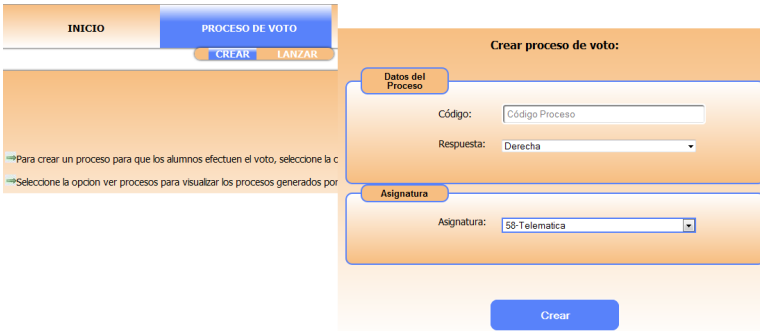

Figura B.5: Creación del proceso

- Lanzamiento de proceso: Se accederá mediante el menú de navegación superior a Proceso de voto -¿Lanzar. Se mostrará una pantalla con una tabla con los procesos de votos generados por el profesor. Este seleccionará un proceso mediante el botón circular de cada fila y pulsará el botón "Lanzar Proceso":

Figura B.6: Lanzamiento del proceso

- Visualización de procesos: Una vez que los alumnos efectúan el voto oportuno el profesor podrá visualizar el proceso accediendo mediante el menú de navegación superior al apartado "Ver Procesos". Al profesor se le mostrará una tabla con los procesos que han sido lanzados y podrá seleccionar el que quiera mediante el botón circular de cada fila. Tras seleccionarlo y pulsar "Ver Resultados" se mostrará una nueva ventana con los resultados del proceso de voto:

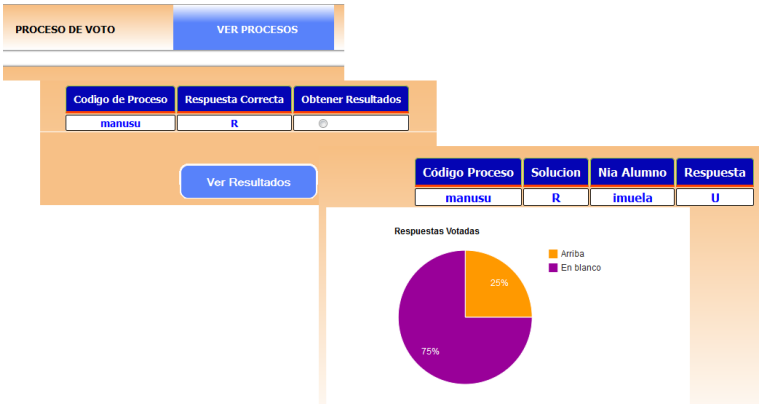

Figura B.7: Visualización del proceso

Apéndice C

Manual de usuario de la aplicación Android

En esta sección se detallará el mecanismo que debe seguir el alumno para enviar un voto relacionado con un proceso.

- Nada más arrancar la aplicación lo primero que el alumno deberá hacer será configurar la IP y el puerto del servidor de aplicaciones del modo "IP:puerto" (Ej: 192.168.1.12:8888):

Figura C.1: Configuración de la IP y el puerto del servidor

- Si tras configurarlo la aplicación encuentra conectividad se abrirá la pantalla para que el usuario introduzca sus credenciales (NIA y contraseña):

Figura C.2: Registro en la aplicación

- Si son correctas, se accederá a una interfaz con un único botón que será “Votar”:

Figura C.3: Busqueda de procesos

- Si al pulsarlo, encuentra algún proceso de voto activo se accederá a una nueva interfaz donde el usuario deberá mover el dispositivo en la dirección que quiera votar. El dispositivo se deberá mantener con la mano en una posición natural, es decir, con la pantalla del dispositivo apuntando hacia arriba y hacia el propio usuario. El movimiento consistirá en un giro de muñeca hacía el lado donde se desee efectuar el voto (es decir, la dirección hacia la que apunte la parte de arriba del dispositivo, será la dirección que este tome como voto):

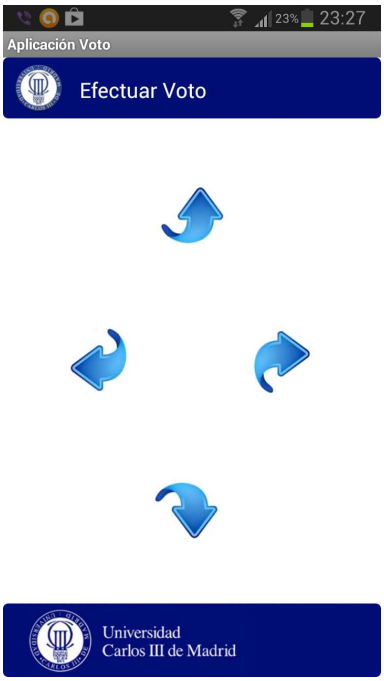

Figura C.4: Ejecución de movimiento de voto

- Finalmente la aplicación mostrará una ventana de progreso mientras almacena el voto en el servidor. En esta ventana se mostrará el voto realizado por el alumno:

Figura C.5: Confirmación del voto enviado

Bibliografía

[1] Fases del desarrollo de una aplicación: *http://es.wikipedia.org/wiki/Desarrollo_por_etapas*

[2] Datos de uso de SmartPhones: *http://e-libros.fundacion.telefonica.com/sie12/aplicacion_sie/ParteA/datos.html*

[3] ¿Que es Android?: *http://www.xatakandroid.com/sistema-operativo/que-es-android*

[4] ¿Que es Android? - 2: *http://myappandroid.wordpress.com/2011/10/23/\%C2\%BFque-es-android/*

[5] Open Handset Alliance: *http://www.openhandsetalliance.com/oha_members.html*

[6] Evolución de Android: *http://mundogeek.net/archivos/2010/03/19/versiones-de-android/*

[7] Sensores en Android: *http://developer.android.com/guide/topics/sensors/sensors_overview.html*

[8] Sensores en Android: *http://www.androidcurso.com/index.php/recursos-didacticos/tutoriales-android/31-unidad-1-vision-general-y-entorno-de-desarrollo/98-comparativa-con-otras-plataformas*

[9] Cuota de mercado Android: *https://www.idc.com/getdoc.jsp?containerId=prUS23771812\#.UV4B4lf7Blk*

[10] Cuota de mercado Android en España: *http://www.kantarworldpanel.com/es/Noticias/Cuatro-de-cinco-nuevos-smartphones-son-Android*

[11] Gramlich, Nicolas *Andbook! - Android Programming*

[12] Plantilla del presupuesto de la UC3M: *http://www.uc3m.es/portal/page/portal/administracion_campus_leganes_est_cg/proyecto_fin_carrera/Formulario_PresupuestoPFC-TFG\%20(3)_1.xlsx*